

THE HOWS AND WHYS OF LEVEL DESIGN

SECOND EDITION

by Sjoerd "Hourences" De Jong
<http://www.Hourences.com>

This book "The Hows and Whys of Level Design - Second Edition" is copyright 2008, by Sjoerd "Hourences" De Jong

By possessing and/or reading this book you agree to the usage terms. See the Copyright and Permissions chapter for more information.

I am an independent author and am not supported by any company, group, or organization. I put hard work, effort, and my personal knowledge into the creation of this book. By stealing the content within, or illegally reproducing this book, you affect me, the author, not a multi-million dollar company.

Please respect the time and effort I put into writing this. If you obtained this book through illegal means, please buy it if it proves helpful. Knowledge is precious and if the wisdom contained in this book can garner you a (better) job, then the small purchase price is worth the money you will gain from putting the knowledge to use.

TABLE OF CONTENTS

8	ABOUT THIS BOOK
9	ABOUT THE AUTHOR
11	INTRODUCTION
13	INTRODUCTION
13	THE BASICS
15	THE HISTORY OF LEVEL DESIGN
16	THE CHALLENGES AHEAD
17	PROFESSIONAL LEVEL DESIGN - OVERVIEW
19	PROFESSIONAL LEVEL DESIGN – COMMON PROBLEMS
23	DESIGN
25	INTRODUCTION
25	THE CREATION OF A NEW WORLD
26	THE CHECKLIST
32	THE FLOORPLAN
33	GROWING PAINS
33	SUCCESS AND EFFICIENCY
34	ORIGINALITY
35	ORIGINALITY CONTINUED
36	CLICHÉS
37	HYBRID SETTINGS
37	CONCEPT ART
39	GAMEPLAY
41	INTRODUCTION
41	BASE PRINCIPLES
43	FLOORPLANS
45	MULTIPLAYER
45	FLOORPLANS

48	ITEMS AND POWER-UPS
50	TRAPS
50	PHYSICS
51	COVER
51	AURAL FIXATION
52	IN-DEPTH EXAMPLES
52	CTF-CORET
54	CTF-LOPO
56	FINAL THOUGHTS
57	SINGLEPLAYER
57	THE BASICS
58	AI PLACEMENT AND BEHAVIOR
60	SCRIPTED EVENTS AND GAMEPLAY VARIATION
62	FLOORPLANS
64	LANDMARKS
65	INTERACTIVITY
66	TRAPS
66	ITEMS
67	DIFFICULTY
68	PACING
69	FINAL THOUGHTS
71	AUDIOVISUALS
73	AUDIOVISUALS
73	INTRODUCTION
74	COMPOSITION
74	INTRODUCTION
74	THE BASICS
76	STAY ON THE PATH
77	COMPOSITION OF NATURAL OBJECTS
77	MOVING GEOMETRY
78	ATMOSPHERE THROUGH COMPOSITION
81	LIGHTING
82	IN-DEPTH EXAMPLES
82	SAE
82	KRODAN
84	GEOMETRY AND ARCHITECTURE
84	INTRODUCTION
84	COMPOSITION AND BASICS
87	UNITY AND CONNECTIVITY
89	CUBISM
89	WEIGHT AND BALANCE
90	THEMATIC DETAIL
91	TERRAIN AND NATURAL LANDSCAPES
91	TERRAIN
92	TERRAIN TEXTURING
94	ROCKS
98	HORIZON
99	THE SKY

102	TEXTURING AND MATERIALS
102	INTRODUCTION
102	BASICS
105	COLORS
109	MATERIALS
110	LIGHTING
110	INTRODUCTION
110	THE SOURCE
112	LIGHTING COMPOSITION
113	BLACK AND WHITE
114	COLORS
116	COLORS CONTINUED: COMPOSITION AND CHOICE
121	TEXTURING AND LIGHTING
122	REALISM AND THE EXCUSE
123	SOUND
123	INTRODUCTION
123	USAGE
124	SOUND VARIATION AND PITCH
125	THE POWER AND STRENGTH
126	IMMERSION AND ATMOSPHERE
126	INTRODUCTION
126	BASICS
127	HISTORY AND SIZE
127	MOVEMENT
128	DETAILS AND CLICHÉS
128	STORY AND DEPTH
130	IMMERSION AND DEPTH IN MULTIPLAYER
132	THE CONCLUSION
135	APPENDIX A - LEVEL EXAMPLES
135	SAE
141	REDKIN
148	KRODAN
153	APPENDIX B - INTERVIEWS
153	CEDRIC FIORENTINO – GENERAL LEVEL DESIGN
157	ROGELIO OLGUIN – GENERAL LEVEL DESIGN
159	ANDREW WELDON – SINGLEPLAYER GAMEPLAY
162	TOM HANRAHAN - ARCHITECTURE
165	DANIEL LUKA - TEXTURING
168	MATTHEW FLORIANZ - SOUND
173	LIST OF SCREENSHOTS
175	CREDITS AND THANKS
177	COPYRIGHTS AND PERMISSIONS

ABOUT THIS BOOK

Level design is the core of a video game and forms the backdrop upon which all the other elements in a game are hung.

It is also, however, one of the most complex aspects of game development. Level design is an amalgamation of a wide range of specialties including architecture, materials and textures, lighting, gameplay, sound, special effects (FX), and more. It is this combination which makes it so complex. Each aspect needs to be controlled individually, and balanced against the others while a consistent overview of the whole is maintained. All the individual elements must remain in harmony with each other and the rest of the level. Producing a good looking and, at the same time, well playing level, is not easy.

This book covers the artistic/visual side of level design as well as the gameplay side. This book does not include technical 'how-to' tutorials. This is an explanation of the 'why' behind level design. Instead of writing yet another tutorial about how to create a virtual area, this will be a larger treatise exploring the reasons behind making that area look cool and play well.

There will be tips, techniques, and explanations of the logic and theory behind design choices. How does the gameplay in a level work? Why does this type of architecture evoke this feeling, or why does it look better in situation A rather than situation B? Why does this color combination work better than that one? What kind of floor plan could best improve the gameplay in this area and why? What kind of sound setup should I use? These are the types of questions that this book will attempt to provide answers to.

The book is divided into four primary parts: Introduction, Design, Gameplay, and Art; and two appendix chapters: Examples and Interviews. Each of the four primary chapters are broken down into multiple parts consisting of a brief introduction, followed by portions dedicated to certain aspects of the overall theme, for example floorplans, sounds, textures, lighting, etc... Each chapter has illustrations portraying examples of the items discussed.

The book, as a whole, is intended to be universal in scope. Most of the information presented can be applied to a wide variety of games, game types, and themes. However, most of the author's experience is with 3D action and/or First Person Shooter (FPS) games, so these will often be the focus of examples and discussions, especially as regards gameplay.

The sources to the many screenshots used in this work are all listed in the image index at the end of the book. Also, some of the screenshots represent levels that have been changed to suit the example better and thus may not represent the actual level.

Some of the examples given may seem rather obvious at first glance. However the difficulty, and thus the reason for the exposition, lies in identifying these 'obvious' problems when they have been interlaced with other situations during development. Spotting them in a clearly drawn example is much different than spotting them 'in the wild' with several layers of, for example, art passes and FX passes applied to them.

The book uses two symbols to let the reader quickly identify good and bad examples. The green symbol obviously stands for a good example, the orange one stands for a bad example.

This second edition corrects some grammar and typos and features a new layout. In addition, it includes an updated cover, dozens of updated example screenshots, and a number of new topics.

ABOUT THE AUTHOR

I began developing my skills in the online *Unreal Tournament* mod scene, in which I remain active today. I am entirely self-taught.

I have worked as a level designer and environment artist on-site and remotely for both small startup studios, and for large established studios. These have included Guerrilla Games, Streamline Studios, Psyonix Studios, Digital Extremes, Epic Games, Webzen, Prophecy Games, Khaeon, and Starbreeze. I have worked on titles such as *Killzone*, *Shellshock nam'67*, *Unreal Tournament 2004* and its *ECE* add-on pack, *Warpath*, *Huxley*, and *The Chronicles Of Spellborn*. For *UT2004* I created two of the most well-known levels: DM-Rankin and ONS-Torlan. Both were featured in the *UT2004* demo and have each been played literally millions of times.

In addition to this book, I am the author of *The Hows and Whys of the Games Industry* (2007) and have also written a few dozen tutorials on various subjects, that can be seen on my own tutorial website. I have also been an Unreal Engine Instructor for game studios Playlogic and Vstep.

As a mod developer I have created dozens of free, stand-alone levels for a number of games. I have also worked with teams on several large mods including *'The Ball'*, *'Operation Na Pali'*, *'Xidia'*, and *'Jailbreak'*. My mod work was also recognized by Epic Games when they awarded me several prizes for my entries into their million-dollar *'Make Something Unreal'* Contest, including first prize.

Level design is not an exact science, and different people always have various opinions as to what is beautiful, what plays well, and what is fun. Some of the content in this book could be considered more subjective than objective, but one can always choose what to keep and what to throw away. However, over the years, I have learned that certain methods do work better than others and that the basics of level design can be boiled down to two basic tenants:

What to do and what not to do. One could, of course, ignore all the lessons and attempt to find out what works best independently, but why do so when one can learn from others' experiences and start with a solid base before experimenting?

The content covered is aimed at those at an intermediate or advanced skill level. A certain basic level of knowledge is necessary before starting this book. For those who have little to no experience with game editors, or the concepts behind how a current video game works, the information will be of little use.



INTRODUCTION



INTRODUCTION

This small introduction chapter will focus on the basics of level design: its history and its function as a profession. The overview will be brief as the goal of the book is to grow a better understanding of how to create high quality levels. However, it's never a bad idea to gain a wider understanding of the subject at hand.

THE BASICS

Level Design is all about the combination of widely varying individual elements. The following, in no particular order, are keywords that compose a level.

- Gameplay
- Floorplans
- Interaction
- Storytelling
- Sound and Music
- Lighting
- Technical Challenge
- Architecture and Geometry
- Textures and Materials
- Effects
- Composition
- Depth
- Atmosphere and Immersion
- Optimization

To master level design is to understand and balance a wide range of components; from architectural design to the technical side of game development. It is a broad subject, and therefore quite challenging to master, especially with the high level of detail found in current video games.

Generally, one could say that a level relies on six 'pillars' to successfully support a certain level of quality. The more of these demands the level meets, the more likely it is that the level will turn out well.

- **Optimization and Polish**

Levels, and all of their composite parts, are rendered by the engine in real time. Therefore, the level must be as optimized as possible in order to run well. If a level does not run well, it is not playable, and therefore is a failure. A level must be playable. The entire design of a game functions on the idea that, when complete, it will be playable.

This pillar also includes polish. A level must be polished, or the player's suspension of disbelief will break down and/or the gameplay will become frustrating. Getting stuck on a tiny little piece of insignificant collision is not fun. Falling out of the game world is not fun and seeing out of the game world ruins a player's immersion. The presentation of a level can make or break the experience. No matter how mind-bogglingly fun the gameplay is or how 'next-gen' the visuals are, if the level simply does not run, or is extraordinarily frustrating to play, no one will notice the quality of any of its parts.

- **Gameplay**

Levels, with few exceptions, are meant to be played. In order to create a good level, a level designer must address both gameplay and visuals – not just one. Great graphics can attract an audience, but that audience won't stick around if the gameplay doesn't hold up on its own. If a level is no fun to play people will stop playing the level, and often the game.

- **Immersion**

The games industry spends billions on graphic development every year. Its spent both in the development of new hardware and software and also on the time spent actually creating the art work. The primary reason why so much effort is expended on the visual side is to enhance the immersion of the game world. If games were solely focused on gameplay, the industry would still be putting out games similar to those from the eighties and nineties. In fact, immersion actually enhances gameplay. The player must believe in the world that the designers present to them. The player must experience it, smell it, hear it, and be completely enveloped by it.

- **Visuals**

As mentioned before, the graphics initially attract the audience. The visuals are the level's presentation. If it looks bad, few will try it. Simply put, bad presentation scares people off. Visuals are also the number one tool in creating atmosphere and encouraging immersion. The better it looks, the easier it becomes for people to associate with the game world.

- **Functional Design**

The design of the level must reflect the story and theme of the world, and enhance the gameplay mechanics. It must also work in harmony with all the elements in the level, in that they must appear to belong and serve logical purposes. A floorplan is created with the game's core mechanics in mind and thus accommodates the gameplay. Its design, the way it's built and why, is a function of the greater elements. Each object in a level must serve and enhance another element within the level and, hopefully, within the game as a whole.

- **The Combination**

Perhaps the most important pillar of them all - a balanced, harmonious combination of each of these pillars makes a level good. A level can have awesome gameplay, and awesome visuals, but if the two do not complement each other it still won't work. It is only the product of the combination of all of these pillars that can ensure a certain level of quality.

THE HISTORY OF LEVEL DESIGN

Level design, as a profession, is still relatively young. At most, it's been around as a distinct job title for about 10 to 15 years. Only since the late nineties has the profession really come into its own because games had evolved far enough that developers had to begin to specialize their teams on larger scales than before.

MUD games, a form of very early Massively Multiplayer Online Role Playing Game (MMORPG), were the first type of games that required an actual form of level design. Most notably, 1991's game *ZZT*, by Tim Sweeney, was one of the first games that allowed players to build their own levels with relatively easy to use tools. A year later, in 1992, level design became even more meaningful as games like *Wolfenstein 3D* and, a year later, *Doom* were released with free level editors. These were arguably the first games that went into 3D level design.

Afterwards, level design grew quickly in popularity as a hobby among video game enthusiasts. This became especially true after 1996's releases of *Quake* and *Duke Nukem 3D* which both shipped with level editors. Hobbyists quickly picked up on the level editors and started designing their own levels. In conjunction with the rise of easily-accessible internet connections and various methods to share these levels, and the knowledge to make them, the level design hobby became even more popular. Finally, developers started to take notice and began hiring hobbyists from the seeds of the game communities that were forming online. Hobbyists began transitioning to professional level designers in a growing industry.

However, it wasn't until 1997 and 1998 that level design truly came into its own. The releases of *Quake 2*, *Half-Life*, and *Unreal* contained an allure strong enough to attract thousands of enthusiasts to spend their time not just in the games; but also delving into the tools that created them. The rising complexity of games, for example the introduction of correctly colored lighting, higher polycounts, and texture resolutions climbing into the 256 pixel range also meant that game engines and computers were powerful enough to render environments that, more closely than ever, mimicked the 'real world'. For the first time, the visual side of the level started to have deeper design meaning. Levels evolved into environments that weren't solely focused on the gameplay, but also on architecture, story, immersion, and other elements now considered integral in today's level design field. The subsequent releases of *Unreal Tournament* and *Quake 3* pushed level design – particularly multiplayer level design – even further. Its popularity grew even more, and more people than ever started spending their free time in designing levels for these games – including me. In the heyday of these two games, along with *Half-Life*, budding level designers and hobbyists released more than 25,000 levels for less than a handful for games. Countless more were included in hundreds of mods that soon followed. Level design had become a big deal.

At that point, more and more hobbyists were being hired by aspiring game development studios and level design slowly began to evolve from a part-time hobby, to a legitimate full-time job.

After 2000, games began to take their first tentative steps into large, open, outdoor environments. 2002's *Morrowind* introduced players to a massive open world. Level design, for some games, changed from a focus on encounters through series of rooms and corridors, to the inverse – encounters dotted around vast open areas. Iteration on successful level design over the following years came from numerous sources such as 2002's *Unreal Tournament* 2003, 2004's *Unreal Tournament 2004*, and *Half-Life 2*, and again in 2008 with the releases of *Unreal Tournament 3* and *Call of Duty 4*.

Over the years, level design evolved from one of the aspects a single person, or very small group of people, would work on, to a very team-based effort.

The first level editors were fairly simple and not very user-friendly. Only after games like *Unreal* and *Half-Life* were released did level editor programs become more accessible, although even these don't compare to today's versions of the same, or similar, software.

In the beginning, level designers worked primarily with Binary Space Partitioning (BSP) as the principle geometry in a level editor. (Currently most games either use 'meshes' imported into the game engine, or a combination of mesh and BSP.) *Doom* was the first game to make use of BSP. All the *Quake*, *Doom*, *Half-Life*, and *Unreal* games use BSP, at least at their core. BSP is a method used to render 3D space very efficiently; thus its popularity during the nineties when hardware capabilities were more limited. BSP has its own set of rules for use – specific 'do's and don'ts' – with specific limitations more restrictive than those encountered when using a 3D modeling application. On the other hand, texturing the surfaces is quicker and, since they are part of the engine, BSP-modeled environments are generally easier to set up and modify later on, compared to environments that are 100% modeled outside of the engine and imported later on. Those early restrictions still influence today's level design.

Level editors have evolved from being a geometry-creation tool to a compositing tool used to bring disparate pieces together to form the game and apply finishing touches. Editors are more focused now on integrating gameplay into levels that are, more or less, modeled in exterior package, touching up the environment with lighting and materials, and applying particles and audio cues. Editors used to be little more than a tool with which to place cubes (literally). Now, however, they contain the ability to preview almost everything in a game in real time. What you see is what you get! WYSIWYG!

THE CHALLENGES AHEAD

Level Design faces some serious challenges in the future of this ever expanding and changing industry.

• Consoles

The rising popularity of consoles and the diminishing popularity of PC games will almost certainly impact modding and amateur level design. One needs a PC to mod a game and, even if consoles offer the same feature one day, they will never be able to match the power and the flexibility of a PC. Modding games needs a PC – after all, the games are developed on a PC to begin with. The modding community, for games like *Unreal Tournament 3*, is typically small but active immediately after the games are released. Compared to the heydays of hobbyist level design around 1998-2002, it's only a fraction of the size it used to be. While level design as a profession will not disappear anytime soon, it's becoming less attractive to get into it through modding. Opening up console games to community development could be an option, but it remains to be seen if this will ever take off.

• Rising complexity

The increasing complexity across the field of level design also increases the difficulty for people trying to get into it. Despite more support available, and more user-friendly tools, the complexity has outstripped these advances which means that the regular hobbyist who just wants to create something fun may find it too complex these days and give up. This rising complexity may very well scare off the majority of interested people. It may also reduce

the general level of quality of community-generated content as the talent pool shrinks in response. This lower quality level design may eventually hinder the development of level designers and modders. The mod community historically has been a recruiting pool; a potential ticket into the industry. If the modding community is unable to keep up with game development at home, potential developers will have to start drawing people from other places such as game development schools. This potential drop of interest from commercial developers could, in turn, discourage community involvement and thus perpetuate the community's decline – a vicious circle.

This rising complexity will also drive up filesizes and associated hardware requirements. What server admin wants to add a 200 MB level to their server? At today's download speeds, this could well discourage community content propagation – especially when downloaded while in game, when every second counts. If an admin can choose between a high quality 200 MB level or a simple 5 MB level, which will be put online? The widening gap in quality may do more harm than good to community-driven level design in the future.

• Fragmentation

There is continual fragmentation between game-specific communities as well as in the actual profession. There are so many games available that people interested in level design become scattered around shrinking communities. Many PC games ship with content editors of some kind, but there is only a limited audience who are both interested, and able. The huge choice interested people have these days, fragmentates the communities.

In addition, the actual profession is becoming overly-fragmented. In the past, just a 'level design' position was called for. Some studios these days have divided this into sub-categories such as gameplay scripter, environmental concept artist, texture artist, lighting artist, and more. Level design, at its core, is about achieving a balance between all the disparate elements. If too many different developers work on a level, the balance becomes easier to upset. Finally, it would be possible to lose the position of a 'level designer' altogether if the fragmentation continues in the industry.

PROFESSIONAL LEVEL DESIGN - OVERVIEW

As mentioned above, 'official' level design positions have recently been split into several positions, depending on the scope/size of the project, and the studio/developer.

This started to occur somewhere between 1998 and 2000 when textures became complex enough to warrant the addition of a texture artist as a position in itself. Previously, this responsibility belonged to level designers. Between 2002 and 2004, more and more studios began to separate visual work and gameplay work into two distinct positions. The new 'visual' level designers were renamed 'environmental artists' and the 'traditional' level design position became focused on gameplay and scripting. This dissemination of level design responsibilities has progressed even further through later schisms. At present, 'traditional' level design has been split into many 'specialties', each of which can command a distinct position at a studio: Lighting Artist, Gameplay Scripter, Floorplan Designer, Material/Texture Artist, Multiplayer Level Design, Singleplayer Level Designer, Cutscene Artist, Water Artist, and more. It is now nigh unto impossible for a level designer to work in a position where they can perform the full range of work that a level designer did less than ten years ago.

It is important to remember that the industry itself never really settled on a specific definition for a Level Designer's responsibilities. Each development studio carries its own list of what is expected from a level designer, and some studios have also moved away from

'level design' as the title of the position. This can, indubitably, cause some confusion when looking for a 'level design' job. One studio may list a 'level design' position and expect the designer to design floorplans, game stories, and design documents. Another studio can list a 'level design' position and expect the designer to script gameplay in a space that's already constructed. The descriptions that studios apply to the title can also obfuscate exactly what they expect from the person who eventually fills the role. Thus looking for 'Level Designer' positions is more difficult than it used to be and requires one to read the 'fine print' of the position's description.

Because of the above, it's recommended that, unless level design is just going to be a hobby, that some sort of specialization is considered. Become familiar with all the various aspects of level design – layouts, scripting, lighting, environments, etc. – but then focus on one or two specific areas. There's no reason to attempt to master all the aspects as that will prove to be too much too learn. Just remember that the aspect you focus on, is the one that's the most enjoyable. At the very least, focus on either visuals, or gameplay.

Level Designer – Focused on gameplay.

- Designs floorplans
- Writes design documents for the level describing the mood, background story, and gameplay highlights
- Places gameplay-related items
- Scripts AI

Level Designer – Focused on the visuals.

- Models geometry (either in-engine, in a 3D modeling application, or both)
- Deals with normal maps and high poly geometry
- Handles materials and textures
- Adds lighting
- Creates particle FX

After working through a few levels, if you find yourself focusing more on the visuals and less on the gameplay, then you might be better suited to being an environmental artist. Generally, it is this author's opinion that becoming an environmental artist is more difficult than a gameplay designer, as far as technical skills are concerned. A good environmental artist needs to know not only the game editor, but also a 3D modeling package, and a 2D program as well, in addition to having some 'traditional' art skills.

If you find that you care more about the gameplay of a level, and deeper strategies or possibilities open to the player, then you will need to learn how to script gameplay. LUA and Unreal's Kismet are good places to start. In depth knowledge of various game editor systems, for example handling AI and triggers, are also important aspects. A gameplay level designer should also acquire a basic knowledge of a 3D application and a 2D application. Clear communication with artists is important and learning a certain amount about their tools certainly helps bridge the gap.

Regardless of which type of level designer you think you are, reading through the entire book is recommended. It's important to understand as many aspects of the profession as possible. Communication with colleagues is one of the most important skills to develop, and understanding where they're coming from can help concepts and ideas flow smoother through the development process.

PROFESSIONAL LEVEL DESIGN – COMMON PROBLEMS

A professional level design position may sound like a dream job but, like any other job, there can be negative aspects as well. Some common problems are:

- **Cooperating with fellow artists or designers**

As level design becomes dispersed among different positions and people in a development company, potential frustrations from cooperation can arise. Levels need to be consistent and scattering the tasks among different people can lead to a fracturing of the overall design. Each person involved may have different opinions or personal views on each aspect of the map. Far too often the designer, or the artist, makes a decision that impacts another party without consulting them beforehand. This eventually can lead to frustration and bad feelings in the group.

- **Managers and leads who are new to level design**

Far too many people in lead positions have a limited understanding of the design issues that are important to a level designer. Thus they can be ineffective when defending level design decisions in times of need, and this can worsen the situation and work against the level designers. Often the 'decision-maker' ends up making judgment calls for design choices designs that they barely understand themselves. A simple example would be of a lead environment artist who has a modeling background and therefore places too much emphasis on modeling and texturing the level. However, because of this, they forget, for example, the lighting or other equally important aspect of the level.

- **Having to acquiesce to others' designs**

Like many professions in the games industry, a level designer is often asked to work on another designer's designs and levels. Even level designers, who are usually expected to create designs themselves, can find themselves with little to no input in actual design decisions, for example gameplay, floorplans, or visuals. Usually the designer can only apply small changes to designs created by people who may not be level designers themselves. Frustratingly enough, it is sometimes possible that these people may have a good understanding of art, or of gameplay, but not necessarily how to execute and incorporate them well. Attempting to persuade them to make certain changes can sometimes be an exercise in futility.

- **Not enough involvement**

Closely related to the previous point is that sometimes level designers are not involved enough in the entire development process of a level and its environment. While level design is a core aspect of a game, the designers themselves are often the people who hear of decisions and alterations last; after every other department has already had their say. This simply does not work well. Level designers are supposed to be familiar with every aspect of a level's design and think beyond the scope of the average modeler or texture artist. Not involving the level designers enough can significantly hinder them from doing their job well.

• Inefficient Designs

Also related to the above are inefficient designs. If the lead designer does not have a good understanding of level design and the technical challenges involved with the particular game technology chosen, they may make impossible suggestions and requests, usually to the detriment of the project. Attempting to implement a non-functional feature in a delicate portion of the design or level, or requesting a situation that's out of scope of the gameplay or technology can result in frustration and lost time – often both.

Despite these potential pitfalls, level design is still great fun. And the right team and leadership can go a long way towards reducing, or eliminating some of these problems. Don't be discouraged by these problems. Every profession has its drawbacks, but knowing about these issues can help an aspiring designer pick the right project and company to work for.



DESIGN



INTRODUCTION

Due to games' ever-increasing complexity and the expanding nature of levels in general, it can certainly be said that levels are not easy to design. Levels, as said before, are combinations of dozens of different aspects, the conglomeration of which render them complex by nature. This combination of complex systems itself requires good design from the start in order to avoid an inconsistent and downright messy result.

Because the different aspects are so interdependent, it's very important not to lose sight of a level's 'big picture'. This chapter highlights some of the issues that can pop up when designing a level, as well as some more minor aspects to keep in mind. The overall design is the foundation for a level. Without a clear, strong design, there is no solid base on which to build the level.

THE CREATION OF A NEW WORLD

The most important part of a successful level is its beginning. The way a level starts will determine a great deal about how the rest of the level will evolve and how quickly. In these days of growing complexity, efficiency and speed are valued highly. Getting off to a bad start or using bad work methods can cost time which is usually at a premium to begin with. Part of starting a good design is foreseeing potential problems before anything is created. By doing this early in the process, a good level designer can quickly and easily modify the design to better fit the available time, workload, difficulty, technical limits, or all of the above.

How one begins a new level is different for every person. One designer may write everything down in a design document while another, like me, just plans it out in their head. The method used also depends upon if one is working in a team environment. Working with a team means that the level's design must be communicated throughout the team which usually means some sort of written, drawn, or quickly modeled design that can be passed around and/or presented. How it's done isn't important as long as several key aspects are kept in mind and the end product is of a sufficient quality. If the technology used cannot create lush jungles, for example, then this must be recognized before starting.

A design should progress only when exactly what is wanted and how to accomplish it is known. Exact information is the key to this. Again using the jungle example, one must know what the jungle will look like, the colors it uses, the overall style, how the player will move through it, if the engine can render thick vegetation, what kind of physics will be involved, and too many more to list here.

To assist in this task, I have developed a type of checklist that is at the base of everything I design. The list compares several key values against each other to see if they are possible and if they should be modified. It also helps define the values better. The list checks to see if the rules of, for example, lighting and composition are contrary to each other and if the goal is possible and what direction to take. This extensive chapter will mostly be about the latter.

A level is complex and it takes increasingly more time and effort to successfully complete one; thus failure is not an option. All the areas that could potentially cause a problem should be identified before starting any work. Once the design process starts it should go smoothly; design dilemmas should not occur or, if they do, should be easily overcome with few modifications to the overall plan. Getting stuck can be very demoralizing and time consuming.

THE CHECKLIST

A level always begins with a goal, a theme, or both. The goal may be that the game requires a medieval castle, or that it's missing an ominous environment, or that the level is to be the central hub of the game.

After identifying the basic idea, certain key information needs to be pinned down before starting the level. This 'key information' will be referred to as 'the keys'. The keys communicate important properties about the level. They are the key words the level is built around and provide more information on the level's requirements.

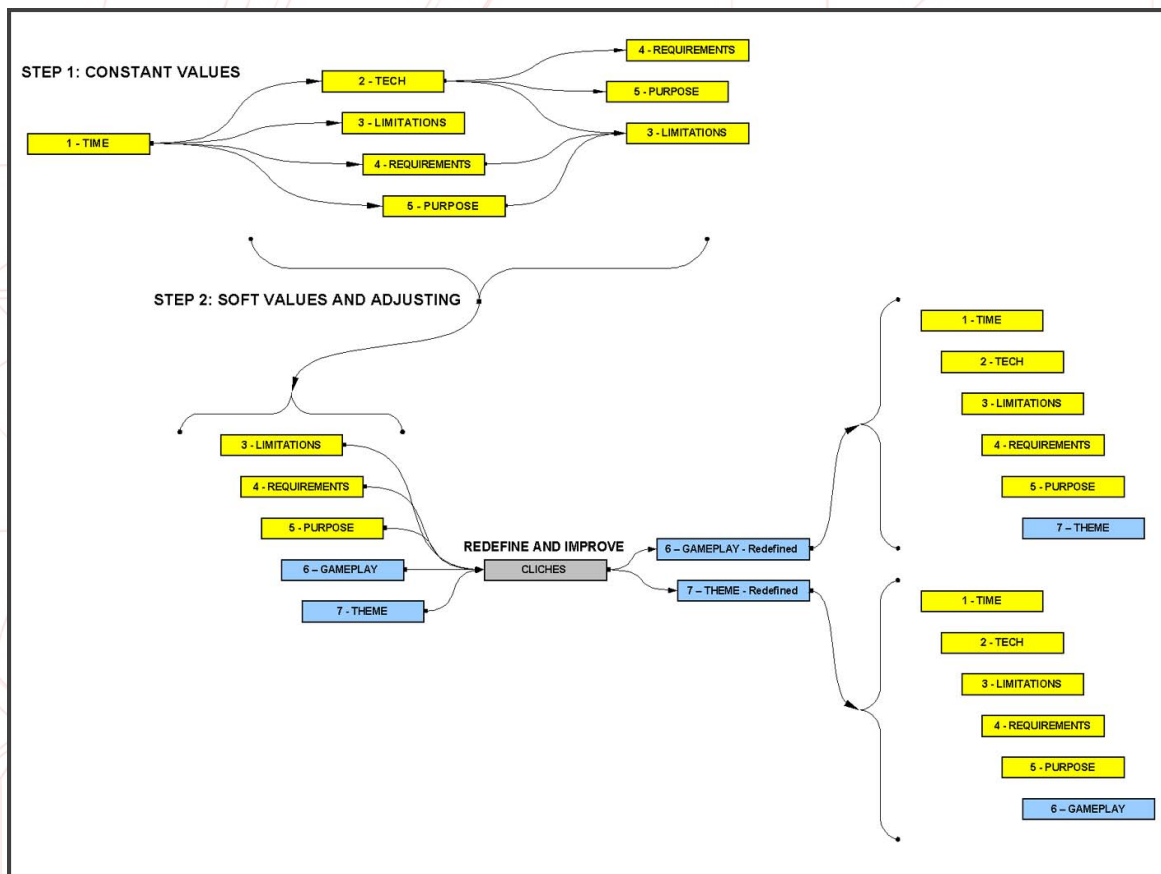
The following are questions to determine the key information for the level-to-be:

- (1-Time) How much time is there available? Is there a deadline?
- (2-Tech) What tools and game engine will be used?
- (3-Limitations) What limitations are there? Is there a shortage of art assets or staff/personal skill limit? Can anything be made or are some aspects beyond the scope of the project because of their complexity?
- (4-Requirements) What kind of requirements are there? Are there any specific elements, for example, special buildings or areas that have to be in the level? When compared to the rest of the game what visual style or theme must the level adhere to?
- (5-Purpose) What is the overall purpose? For example, is it a multiplayer practice level or a singleplayer boss arena?
- (6-Gameplay) What should the gameplay be like? How should it be played? Should there be enough room for a large boss encounter? Or does it need to be large enough to contain a large number of enemies attacking the player? Perhaps it's a vehicle level? Or it is a stealth level? And so on.
- (7-Theme) What theme and/or style will the level have? Will it be a castle or a jungle? Will the style be cartoonish or realistic?

This is all essential information for a level. The order of the list is not as important as the answers. Once the essential elements of the level have been identified it can be run through a checklist to see if it holds up. Will it work? Look right? Play right?

The keys provide the information while the checklist determines if it is possible or not. The checklist combines two or more keys in order to determine if they fit together or not. If the desired theme is a jungle, but the engine can't handle rendering dense vegetation, then these are two keys that do not fit together and the design will need to be adjusted accordingly. This is the type of information the keys provide: essential information that design decisions can be based on before actually starting work on a level. Thinking ahead is the key to success.

The checklist itself is a system for asking questions and making comparisons. The questions are different each time, but the comparisons remain the same. Verify that the individual elements compliment each other; for example the time against the theme and its necessary workload.



There are two types of keys: yellow and blue. The yellow keys are constant values: Time, Technology, Limitations, Requirements, and Purpose. These are the given values and thus can't be easily modified. They are constants. Problems between two constant values are the worst because they will require the largest changes in design. The first part of the structure compares all the constant values against each other to find out if they are complimentary or not.

Time is the only constant value that is checked against all the others. If there is very little time available, then solid Technology and Tools are necessary. If Time is checked against Technology and the determination is that the efficiency needed is insufficient to complete the work in the given Time then there is already a problem.

This also applies to Limitations, Requirements, and Purpose. If certain required elements are too complex to complete in a short time span then that's another problem.

After completing the Time checks, the Tech is then compared to Limitations, Requirements, and Purpose. If one of these requires a feature the engine/Technology can't provide then another problem is identified. An example of this could be the dense jungle which the engine can't render efficiently, or large open battlefields with thousands of soldiers that a console or computer may not be able to handle.

Lastly there's the check for Limitations against Requirements and Purpose. While this may not sound logical it is worth verifying. Sometimes a feature is asked for that simply cannot be completed because of the inherent Limitations. The later examples will illustrate that.

The constant values are those that cannot easily be altered unless a single person is in control. In almost all other situations mod teams, and especially in commercial studios, these constant values are controlled by other people and thus quickly altering them might not be simple. For this reason those problems are the most serious.

On the other hand, the blue values, Theme and Gameplay, can be modified as long as they stay aligned with the constant values. If the Purpose or Requirements require a forest theme, it will eventually become impossible to convert the area into a factory. Instead, if identified early enough, the forest can be redefined so it better suits the level's needs. A forest can still be transformed into a dozen different kinds of forests, which brings up the second purpose of this checklist structure. The checklist is not only meant to determine if something is possible or not, it is also meant to further the development of an idea. When the theme that's given is a forest, or the gameplay is a boss fight, there is still very little detailed information regarding exactly what to make. Before anything is started the more detailed information on what is to be done is still needed. As mentioned before, the forest theme can head in a dozen different directions. The colors, atmosphere, overall styles, landmarks, special features, exact locations, and so on that are to be used have yet to be defined.

The second step of the structure accomplishes this. This step combines Limitations, Requirements, and Purpose with either Theme or Gameplay and creates a redefined Theme or Gameplay from the combination. In order to further develop the Theme or Gameplay situation, its context must be examined. Random ideas aren't always the best solution.

If the requirements state that the theme or gameplay must be part of a scary story or game, then it should be clear that the requested forest or boss fight should be very scary.

Theme and Gameplay also influence each other. If the gameplay calls for the area to convey an atmosphere of fear, but the Purpose or Requirements contain no information on the type of atmosphere then there's missing information. The theme and audiovisuals in general should reflect the frightening situation, and therefore the Purpose and Requirements should as well.

The same situation is also true in inverse when the theme has atmosphere X or situation Y; the gameplay needs to reflect that same situation or atmosphere through, for example, enemy placement and events.

A more clearly defined picture of what to create takes shape by taking a look at all the information given and the relationships that are shared within the different elements. This is the hardest step and is what the rest of the book attempts to make sense of.

In order to redefine these elements, use clichés. As will be explained extensively later on in this chapter, clichés are very powerful. Thus, using their potential of instant communication is a wise move. When something – anything – is mentioned, the first thought one has is usually a clichéd version of the item or situation in question. Think of “depression”. What are the first, and therefore strongest, visual cues that communicate “depression”? Imagine what a child would draw if asked to color a picture of ‘spring’ or ‘summer’. Apply this method to the theme or gameplay and the different available elements. If the design calls for a scary Theme, then think of the clichés that communicate “scary”. What environmental elements convey fear; what colors, sounds, effects, etc. With this new information Theme and/or Gameplay must be iterated on until a clear picture is presented of what is wanted and exactly how it will look and play.

This revised Theme or Gameplay is then again compared to the full list to verify that everything is still possible. For example: Imagine that the revised theme is a dense jungle but the technology cannot support it. It is important to check this after redefining them because the original theme might have been too undefined to allow for a decent comparison. Exact information is important, as will be reflected in some examples further down.

If the Theme or Gameplay manages to get through to the last checkpoint then the path is clear. If it doesn't, and new problems arise, then one or more aspects need to be changed, sometimes a little, sometimes significantly. Reiterate on either the Theme and/or the Gameplay until the problem is solved. If a solution can't be found, then the design is flawed. This will become clearer in the later examples.

The checklist structure may look very complicated and the choice to use it or not is up to the reader. It does take time to practice and master seeing potential problems ahead of time. I, personally, can run any idea through the checklist within a few minutes, but I've been using it for quite a long time. In the end, the whole point is that some sort of system to identify potential pitfalls should be in place before work on the level is started. If another method works that is fine as long as potential issues have been identified and addressed and the design functions well before work is started.

Not only does this list benefit the unity of the design, but it also ensures the practicality of the ideas/goals. Each individual element should be complemented by the others. In this scenario, the architecture is sharp and angular because the overall theme is 'scary'. Likewise, the place feels 'scary' as a response to the style of the architecture.

Determining a visual style is impossible without first understanding the gameplay and the atmosphere. Knowing what is, and equally, what is not possible both describe what is to be accomplished and helps avoid future problems. A designer must have a general understanding of many aspects either in the head, or on paper: how it will look, colors to use, atmosphere, gameplay, lighting, required material/texture and mesh assets, time to create the appropriate assets, and their corresponding difficulty level to implement. The last thing a designer wants is to reach the end of a project and discover that an important piece of the puzzle is out of balance with the work as a whole. This is a large potential time sucker and can also be bad for the designer's motivation.

If the checklist is applied to a few examples, it quickly becomes clear how it can work and why most rules and tips explained in the following chapters are so important. The following examples contain both positives and negatives from commercial, and personal, projects.

• The Dead Forest Example

- Basic idea: Fight a large boss that must be lured out of its nest, led through a forest, and onto a magical seal.
- Key 1 Time: No defined time limit but the faster the better.
- Key 2 Tech: Unreal Engine 2.
- Key 3 Limitations: Very few new assets – recycle existing ones.
- Key 4 Requirements: A forest with a path and a magical seal. Must have a cartoon-like style but also be unique to suit the theme. Needs to occur in an area where a city will be built later on and the initial settlement must be visible.
- Key 5 Purpose: A boss fight where the player must lure the boss monster onto a magic seal to trap it.
- Key 6 Gameplay: A boss fight, beginning with the monster chasing the player.
- Key 7 Theme: A forest.

Summary: The time limit did not conflict with Tech, Limitations, Requirements and Purpose. Nor did Tech cause any problems when compared against Limitations, Requirements and Purpose. The game had plenty of forests already so the tech could surely handle this. Also, because the game already had so many forests, my experience with creating them also meant I could easily make another in a short period of time. This also enabled me to re-use most of the assets. The constant values clearly passed the test.

The 'soft' values of Theme and Gameplay did need to be redefined. Because I, personally, was not responsible for the gameplay in the level, my sole focus was redefining the theme. The basic layout of the map was already in place, since a level lay out was already designed for the city that would one day be on the same spot. The only real question remaining was how to style the forest and the remaining space. I ended up creating a dead pine forest for several reasons: there were already pine forest assets so it was relatively easy to adjust them to a more 'dead' forest. This satisfied the Time and Limitation keys. In addition, the dead pines were a unique theme as they had not been used anywhere in the game and they contributed to a deceased, scary environment. This setting perfectly fit the style goal

because the boss was a big, frightening monster who spent his days eating villagers from the nearby settlement. The cliché was intended to be a dead, dry, and brown environment that reflects the scary, human-eating monster who also had red and brown skin. The design completely succeeded in balancing the themes and was consistent in doing so.

• The Thawing World Example

- Basic idea: A very large, thawing ice world.
- Key 1 Time: As fast as possible.
- Key 2 Tech: Unreal Engine 2.
- Key 3 Limitations: Few new assets.
- Key 4 Requirements: A very large thawing ice world with pieces of thawing jungle.
- Key 5 Purpose: Expanding the game world and providing lots of new, precious space.
- Key 6 Gameplay: Exploration-driven gameplay. Free-roaming, open world.
- Key 7 Theme: A thawing world.

Summary: Time didn't pose any problems for any of the keys except for the Requirements. A thawing world is an 'in-between' situation. It is a combination of two themes or a conglomeration of multiple situations/themes. Such situations are much more difficult and thus more time consuming to create. As there was little time available, it was not a very good idea to pursue such a theme. Also, if Tech is compared against Limitations, Requirements, and Purpose, one notices that creating a thawing world with a frozen jungle is quite difficult. Plants encased in blocks of ice are not created easily, especially in a visually pleasing way and the engine may not be able to render them well with the required multiple overlapping transparencies. This means they would also likely be performance hogs because of all the transparent surfaces both in the ice, and also potentially, in the plants. Finally, a thawing world also requires many new assets as it is such a specific and special situation. All these conflict with Limitations and Time. By now it should already be apparent that this idea is seriously flawed but we will continue for the sake of the example.

Similar to the former example, the gameplay was already in place so I only had to focus on the Theme. The theme description contained very little information that could be of help in designing the style and atmosphere of the area so I was pretty much free to design anything I wanted as long as it did not conflict with any of the remaining keys.

If I think of the clichés of a thawing world I, personally, think of spring, small creeks and ponds, dripping and melting ice, rivers, grass, flowers, and so on, with plenty of blue lighting with some yellow lighting mixed in, and perhaps some orange counterpoints. In other words, fresh but desaturated colors. However, when that clearer view of the environment is checked against the keys several problems crop up. First of all, it is near impossible to create many creeks, ponds, and small rivers with the technology used on such a scale and certainly not with so little time available. Without the typical ponds and small rivers, as well as all the dripping - which is also a tech problem - there would be barely anything left that is recognizable as "thawing". Basically it would end up being just a field of snow with some patches of grass, and a few flowers scattered around. This is a serious problem because games are not precise enough yet to show such subtle situations. Any situation and/or element must be exaggerated if the player is to notice them. To most players such a world would not look or feel like a thawing world but more like a regular winter world. The flowers and patches of grass are elements that are not powerful or large enough to bear the entire weight of the theme. The theme would fail to represent a good looking and credible thawing world.

If the features that must represent the theme or situation are not significant and strong enough they won't be able to push the level in the needed direction and the result will likely fail.

A possible fix could be to create more theme-specific elements/landmarks in the world but the Time and Limitations in this case didn't allow for that. The design was impossible to implement without serious changes. In the end the people who provided the initial key information agreed to withdraw their request for a thawing world which was the base of all

the problems. It was simply not possible with the limited time and assets. A fully winterized theme was eventually chosen over the thawing world and that did manage to get through the process without any scratches.

• The Redkin Example

- Basic idea: An industrial environment with great gameplay.
- Key 1 Time: Unlimited time.
- Key 2 Tech: Unreal Engine 2.
- Key 3 Limitations: Geometry must be relatively simple.
- Key 4 Requirements: Industrial style, moving elements, impressive architecture, high areas, reflects elements of its 'brother' DM-Rankin, offer a fresh visual style yet typical Unreal.
- Key 5 Purpose: Create an industrial building which has a high level of quality across all aspects; from gameplay to visuals to sound.
- Key 6 Gameplay: Hardcore gameplay with plenty of depth and strategic possibilities.
- Key 7 Theme: An industrial map.

As the Time was unlimited, there was no need to compare Time against the other keys. The Tech also posed no problems in terms of Limitations, Requirements, and Purpose. Also, Limitations did not conflict with anything either.

When I combined the keys with the soft values Theme and Gameplay, the result was an industrial environment, a large exterior canyon, many moving elements, and a color scheme incorporating deep orange hues with some blue highlights. Industrial, as a theme, can be very open to interpretation. It can be stretched in many directions and it's possible to incorporate many disparate elements into it, for example moving parts. On the gameplay side I was also able to add many strategic and fun options for the players.

All the aspects of the proposal were possible and they all passed the checklist process. A more detailed description of the various decisions and the reasons behind them can be found at the end of the book in The Redkin Example.

As should be apparent by now, each aspect of the design process is related to the others. Using yellow light simply because it happens to be a favorite color, without referencing any other aspects, is an example of a 'bad' reason. Everything is interconnected. Use yellow light only if it fits thematically and if the choice can be reasonably explained. It must contribute to the big picture. Personal preferences for certain shapes or colors are irrelevant if they conflict with other elements!

I, personally, dedicate a lot of attention to the "Can I do this" question. I once started an Unreal Tournament 2003 level, DM-Teapot (seriously!), and it ended up as one of the few levels I did not finish. It turned out to be huge rather than small, as was intended. The level contained an exterior gameplay area, as a 'roof' over the interior, and an interior area as well. I did not plan ahead and review that this situation could result in serious problems. The inside was under-scaled and therefore too small to allow comfortable player movement, while the exterior contained too much space and was too open. I couldn't rescale one without making the other worse. In addition, I used very large meshes and the Unreal Engine build I used couldn't handle large meshes well at all. As a result the level was buggy and ran badly. Last, I wanted to lightmap all the meshes, which was not a feasible expectation. Had I examined the level with the 'Can I do this' question ahead of time, I would have researched large mesh use and lightmaps and discovered that there were many technical and workflow problems with each. In the end, the level was buggy, ran slowly, and I ended up dropping it because it was unworkable. It also required skills that I did not possess at that time.

I had a similar experience with Ortican, a singleplayer level for the original Unreal Tournament game. I made it larger than both me and my computer could handle which caused such frustration that I almost abandoned the level. It took months to finish and the end results were still only mediocre, in my opinion.

Unlike what some think, the amount of time spent on a map or project does not equal a good product. Experience has shown me that most time spent on levels that took ages to make was usually spent correcting initial design flaws and work around issues, rather than spent on the actual level. Working too long on a level without seeing the expected result can reduce the motivation to work on it because the reward of seeing the creation come to life is perpetually delayed. If work is disrupted due to brainstorming ways around issues, or how to fill up an area, or how to resolve a certain design issue, then there's a workflow problem. This can occur when one is too critical of their own work, or perhaps they simply do not have enough experience to know how to deal with more common problems. Perhaps the design itself is flawed, too difficult, or even impossible to achieve. This last – impossibility – should never occur.

In the end, any idea can be turned into a level or a game but some truly are easier to create than others. Depending on personal experience, available time, and more, the right decision at the right time is critical – even if the decision is to not attempt it at the beginning of the process. The right decision can be the difference between success and failure, dissatisfaction and the reward of a polished result.

THE FLOORPLAN

While floorplans are extensively explained in the Gameplay chapter they are also an essential part of the design phase.

Floorplans are not just about gameplay, they also convey information regarding the visual elements; what object is where and what type of environment area X or Y has. Floorplans also help define the visual composition of an area and are also important for workflow and technical requirements. For example, if modular content does not permit the creation of corridors with stairs then either the floorplan or the modular content must change. This is also true when planning optimization, for example where to place occluders, as well as the general layout of important areas. An important building shouldn't be placed on a whim - the visual composition must be laid out in an interesting way to ensure that the building really does look important. This also applies to the buildings around it; their placement must complement the important building.

A floorplan is more than just gameplay. This is one of the reasons why I do not care for gameplay designers planning entire floorplans without either basic art knowledge, or input from experienced environment artists.

There are several methods for designing a floorplan. Some people swear the only way is to draw them. But the only true way is one's own. Drawing, creating low-poly versions, or even using Legos does not guarantee success or automatically improve the quality of the design. It's simply a method to assist the designer. I personally always use just my head. However, some people are more comfortable with paper, or a whiteboard. Whatever works is good as long as the important information comes through.

When working with a team, however, it may be useful to do actually draw a floorplan at some point. A drawing is still the best way to convey an idea to other people. In such situations I use Photoshop and add pictures near certain key locations to indicate what type of area I have in mind. These pictures come from other games, movies, or through image searches. The pictures can quickly show co-workers what kind of atmosphere, style, gameplay, or situation I imagine for an area.

GROWING PAINS

Many people who have only just started level design, those with less than two years of experience, often get stuck because of a lack of ideas: designer's block. The more experienced one becomes, the faster good ideas will flow. Coming up with good ideas also requires practice. Notice the word good. Good means the idea is not only possible but also shows plenty of potential from the outset. As more experience is gained, the faster potential problems can be identified and avoided, and therefore the quicker the best ideas will rise to the top.

For a beginner, there are many aspects of design to prevent getting stuck. First, get excited but keep it simple otherwise the 'Restart Syndrome' may appear. This occurs when a level is started over and over again without progressing because the author is either overly critical or because they lose motivation because of technical or artistic issues, or both. Many people experience this and it usually leads to naught. One of the most important aspects to being a successful level designer is having the strength to see a project through to the end, even when it takes months. The amount of time one can work on a level before becoming disenchanted or losing focus is a trait that is developed over time. Thus, start simple, and gradually add in complexity as you go. Often, beginners have little idea of how long certain aspects of level design take to complete or the level of complexity certain goals can require. Many times they pick very difficult themes or large concepts, such as complete cities or alien palaces and therefore fail to finish them. Knowing one's own capabilities before beginning a project is a necessity.

If one is completely stuck, or unable to complete a level, then copy an already successful design. Forget the rules and the desire to create an individual style or any attempt to be original. These are the traits to think about only after gaining more experience because it will only make things more complicated at the start.

As a beginner it's possible to learn plenty by copying other pieces of levels. Pick well known and simple themes, like a castle, so that plenty of examples are available. Examine them thoroughly and, if still stuck, look at the examples again to see how the architecture was created, how the lighting was implemented, or whatever aspect is causing the roadblock.

Don't be too critical of the work. Expecting a first level to be just as good as something a person with five years of experience made is a mistake. Always strive for that level of quality but forgive yourself some faults; be realistic and patient with your expectations or else your motivation will evaporate. Allow yourself to be proud of what you make, even if it's quite simple!

There are millions of places to find inspiration in the event designer's block crops up. Movies, books, comic books, and other games, of course, are prime locations but a random picture search on the internet can also do miracles.

SUCCESS AND EFFICIENCY

You, or the people you work with, may want you to try to 'stand out'. This is especially true for amateurs working on mods or those working on low budget games. To stand out from the crowd, you must be able to show a project that will grab and hold people's attention. This can be done either through quality, style, story, atmosphere, or gameplay. Have a look at the amount of user-created levels the Unreal community has produced over the past several years. More than 10,000 user-made levels have been created and released and that number is potentially even higher for games like Half-Life. The same is true for mods and full games. Your personal level is one of tens of thousands. If it is to be noticed, then it must have something special. Half-Life 2, for example, suffers from too much pre-fabricated content. There are hundreds of levels that all use the same theme: City17. Unless a level has

either a new theme, or a new gameplay element, or pure quality and polish, no one is likely to notice another City17 level.

Regard the level as a product on the market. Ask yourself what the selling points would be. Why would someone be interested in this level and not one of the thousands of others? What's so special about this one? What's going to grab people and persuade them to download and play it?

I always try to reach as wide an audience as possible. I do this by ensuring that my levels have multiple selling points. If I put all my effort into just the sound, then people interested in gameplay or graphics would probably pass it by. By addressing many important features a wider audience can be reached.

To aid me in this goal, I use my 'minimum effort, maximum result' theory. Efficiency is an important asset as complexity increases in games, and the tools used to create them. A game or level that's huge or has many special elements is possible, but would take ages to create and most people and companies are not in a position to do so. Successful levels and game concepts are often very simple when peeled back to their core. Ideas that are simple to create, but generate maximum impact, are the goal: minimum effort, maximum result.

History has proved that complex concepts do not always guarantee success. The formulas used to create Couterstrike or game modes such as Deathmatch and Capture the Flag are very simple at their heart. But, at the same time, they still offer gameplay depth inside the simple core concept. The same goes for *Portal*. Its levels are deceptively simple and the entire game uses fewer than 50 textures, yet it's still a great game and the visual and auditory presentation is captivating. Now that's efficiency!

It's possible to design a large, story-rich game that takes place in dozens of large environments, and work on it for years, and still have less success than a product that's a simple, soccer-inspired game with a few nice, simple environments that took very little time to put together. This is also true for audiovisuals. While dozens of unique meshes and textures could be created for a scene, they don't always have to be. I, myself, have created entire cavern levels with fewer than 10 meshes and 5 tiling textures. The end result looked better than other similarly-themed maps. What is done with the available tools is more important than the variety of options given by the tools. A few ugly, low-poly models can look better than fancy high-poly models if the latter is used badly, but the low-poly ones receive the correct attention with cool lighting applied, a perfect composition, and a great atmosphere. As mentioned before, any idea can work, but some truly are better than others. Pick the right ones!

ORIGINALITY

While originality is important and can help a project stand out it is never the only aspect a good level requires. Designers sometimes become so focused on trying to be original that they lose their focus on other important elements. Originality means little if other aspects are horrible, for example the lighting or the gameplay. Developing originality always requires a solid base on which to build, and this is where originality often goes awry.

Usually, the problem arises from bad priorities. Without a stable base, original creations cannot be built. Design and create the base first, and then grow the rest into something original.

For example, say a designer decided to start a level with the ultimate goal of using purple lighting as an original theme. By deciding this, they have skipped multiple, vital steps and are completely missing the solid base upon which the environment is built. Purple will probably not complement any other color in the level, or match the atmosphere. Therefore it is important to avoid making random design decisions without considering other aspects it will eventually mix with. Adjust the gameplay and colors according to the theme and purpose – not simply to be original. The gameplay and visuals must co-exist and enhance the

essence of the whole product.

A 'bad' example of this would be beginning a level with the idea of using red lighting everywhere because one feels like using it and because it's original. This would not be good because, later in development, the realization is made that the environment must be a very cold or wet area where an overabundance of red does not fit. Decide on the goals and setting first, rather than making choices solely based on originality. A much better way to convey originality is to find an original way to bring across a chosen theme or setting. Create the base first, and then execute originality over it.

Once, I worked on a project where the design document stated that the level was to be a large and ancient catacomb which was going to be the base of operations for some enemy bosses. Based on this idea, the environment should feel old, scary, and convey a kind of majestic feel due to the presence of powerful enemies. As described in the checklist, this would be the starting point in order to determine which colors would be used. However, the designer tried to skip this step. He wanted the world to be light with white and purple and argued that this choice was good because of its originality. The combination of colors didn't fit the atmosphere. White and purple lighting isn't what one connects to a scary, majestic, cold environment. Also, as will be explained in the Lighting chapter, white is a very difficult color to use. His decision was solely based on originality and not on the base requirements: good lighting and matching theme, style, and atmosphere. Originality should never break the rules of design. There is a reason such rules exist and ignoring them will often cause the level to look unfinished or disjointed rather than original.

Often, there are reasons why certain elements are used more than others. Often, little-used aspects simply don't look good or play well, for example pink lighting. Using these elements will not make a level original. It will usually just make it worse for most situations. Attempt to ensure that original aspects of design aren't countered by a decrease in quality of other aspects.

I am in no way trying to convey that originality is bad in and of itself. I am simply advising that incorporating original elements into levels should be done cautiously, without blind belief behind it. Originality is not a path to success. Make sure there is a solid base for it to fall back on.

ORIGINALITY CONTINUED

Although I may come across strongly against originality, I certainly am not. I believe originality can be a great addition to a level as long as it is solidly supported by the other elements in a level. Originality can actually be considered one of the requirements for success. There are thousands of games out there and an even greater number of mods and levels. Getting noticed and being able to stand out from the crowd is becoming more difficult each day. A standard castle- or factory-themed map simply will not stand out by itself anymore. Somewhat uncommon elements are needed in order to impress the audience. These elements could be as simple as a high level of polish or finely tuned gameplay, or as elaborate as a strong atmosphere, actions and events within the level, style, or even originality. Ask the question 'What sets this level apart from the rest?'. 'Why would someone choose this level over some other random level?'

Dare to dream. Depending on the style and theme of the game, there may be much more possible with a castle theme than what has been done in the past. Instead of a standard European castle, it could be a more original Arabian castle, or a colonial one. Perhaps a thunderstorm or tornado effect could add the extra touch. Perhaps it could be transformed into a haunted castle with a strong atmosphere, or maybe even a combination of all of these.

It takes a special something to get noticed. 'Selling Features' are needed for a level, but they should always be constructed off a solid base.

CLICHÉS

Level design, for a large part, is about clichés: instantly recognized aspects. The reason behind this is that games lack the much required complexity to communicate finer points. The level of detail and technical possibilities are rarely good enough to display specific details and features. In addition, the gameplay is often too fast and distracting to allow players to see small details. What players do see should be direct and to the point. What something is intended to be or represent should be communicated to the player as quickly as possible in as clear a manner as possible. Clichés are communicated through symbols. They are the most easily recognized, the most powerful indicators, that a level designer has in their toolbox.

Think of 'scary' and a few thoughts may immediately come to mind. They could be a full moon at night, light shining under a door, ravens, and more. For every setting, there is a whole list of thematic elements that are commonly associated with feelings and moods that match the setting. It is the designer's task to translate the desired theme and atmosphere into a series of clichés and build upon them.

Clichés are clichés because they so good at communicating emotion. Light from under a door didn't become a cliché by accident. It is used so much because it works so well. Clichés are the most powerful representations of their kind. Ignoring their potential would be foolish.

When designing a level, try to think about the kind of clichés that match the theme and mood. Without strong identifiable markers, many people will not understand what is being shown, especially in games with limiting technical requirements. I'll use one of the checklist examples to illustrate this. As described above, I was once given the task to create a thawing world, but ended up running into many problems. When I need to make a specific theme or setting, the first thing I do is to think of what elements represent that desired theme or mood. In this case, the first words I thought of were spring, birds, flowers emerging, patches of snow, melting ice, water, creeks, rivers, and grass. The problem, however, was that due to time and engine constraints, creating lots of water visuals, like creeks and rivers, was not feasible; nor were the birds. Basically, the only thing I could make was the grass with some flowers around and that simply was not enough to carry the burden of the entire theme for the map. When people would play the level and see patches of snow they would probably think of winter or, at best, spring, which is not quite right. Snow is a much more powerful cliché than patches of grass and flowers. People identify with the most powerful element in a scene to base the rest of their conclusions on.

A possible solution was to drastically reduce the amount of snow so that people would be more likely to notice the grass and flowers but even this was not enough. The level would still appear a little too ordinary to communicate the melting world theme effectively. Also, the trees had to be pines due to asset and time limitations, and pine trees are commonly associated with winter so there was no way the trees could assist in the cliché communication.

In the end, I pushed the level towards a full-scale winter theme and saved myself, and the team, stress, effort, and risk, and was therefore able to implement a much stronger, more easily achieved theme at the same time. This example not only illustrates the use of clichés and their potential complexity, but also shows how the right choice can influence the production.

HYBRID SETTINGS

Hybrid settings are even more difficult to create especially under tight deadlines. A hybrid setting can be defined as a theme or atmosphere that is a combination of two or more. As previously described, an example of a hybrid is a theme somewhere between winter and spring. Another example could be a late morning theme – it's neither morning nor noon.

Hybrid settings can be troublesome because they rarely have clichés of their own. Even if they do, they're often very few and potentially not as powerful, or perhaps they borrow too heavily from each of the two themes that are being mixed together. A late morning scenario, for example, would include some songbirds, a sun that is partially risen, and a well-lit world. All of these features, however, belong to either morning or noon – none of them specifically cater to 'late morning'. The result would be that people playing the level would only notice some of the most powerful thematic elements and thus immediately associate the overall theme and world with those. If the sky is too bright, then the player may ignore the birds and think of the place as a noon setting. The same problem would occur in a thawing world. The snow, or lack thereof, would tilt the player too much toward either winter or summer, thus invalidating the thawing world theme. Unless there's plenty of development time, slowly-paced gameplay, or a powerful engine, the clichés would ultimately be ignored by the player.

CONCEPT ART

Concept art is a fantastic tool to aid a level designer in visualizing the world. While the goal of concept art is to influence, guide, and inspire artists, it can also obstruct them if the art is interpreted incorrectly. Concept artists often have little knowledge about the game engine and how it works. I, personally, have seen this happen many times. For example, concept artists portray large open areas that the engine isn't capable of running. Also, the same open areas would not work well with the gameplay because there's no place for strategy. And sometimes the amount of assets, or time available is insufficient to fully flesh out the space adequately.

It is important to remember that concept artists are not level or gameplay designers. While the vistas they create are often very beautiful and interesting, they often must be adjusted to fit the technical or gameplay requirements – or both – and this is part of the designer's job. Don't blindly copy the concept work – try to look at it critically.

While a cubic room might look great in the concept art, in black and white, or at a specific angle, that does not mean it will look good in game. In fact, for this specific example of a cubic room, it will most likely not look good and the level designer should evaluate the art critically in order to identify its weaknesses before continuing any further.

Adjust the concept art when translating them into levels. Concepts can only compensate for a lack of personal experience or creativity to a certain extent. One needs do more than copy the source. Concept art exists to inspire – not to provide blueprints.

I once saw an artist model a high-tech room based on concept art. The engine he was creating it for had very low-poly technical requirements so the artist was forced to drop most of the architectural details drawn in the concept art. Because of his experience, or lack thereof, he ended up deleting the core elements and focused on secondary, less important aspects. For example, the room in the concept art had six corners and the level designer translated this into a four-cornered cube. This produced a boring and standard room even though the original concept looked great. The designer made the wrong decisions because he did not understand why certain shapes in the concept should be in the level, and why other shapes were less important.

If you do not understand what you are doing, concept art is not going to save you.



GAMEPLAY



INTRODUCTION

Each game in each genre has its own gameplay. No two are 100% alike, but many share the same base. This portion of the book explains the base principal of gameplay that each game needs in order to outline where the 'fun' comes from. This section is divided into two primary sections: Multiplayer gameplay and Singleplayer gameplay. They will each cover many subjects including, but not limited to, floorplans, events, traps, item placement, AI, and more.

Please note that even if your interest lies only in one of the two gameplay aspects, it may still be useful to read both chapters since many of the topics can be applied to either.

BASE PRINCIPLES

At a fundamental level, there are two types of gameplay in a game: Map Gameplay and Core Gameplay. Understanding each is a necessity. Many people think of gameplay simply as 'gameplay' but this does not sufficiently describe the Core game mechanics in reference to the player tools, nor does it adequately address the environment, or map, that the player is dropped into in order to leverage the tools.

• Core Gameplay

The Core Gameplay is the set of gameplay rules and player toolsets provided by the game itself. The Core Gameplay determines how the player will spawn, how fast they can move, if the player can jump, and if so how high, and how the player is able to interact with the world. It also defines the player's goals in the world and the means to accomplish achieve them.

A level designer has little impact on the Core rules. They already exist and cannot be easily modified by the level designer. A level designer's job is to accept the Core Gameplay and then adjust the Map Gameplay to leverage the Core Gameplay.

• Map Gameplay

The Map Gameplay determines where the player will spawn in a world and then gives the player the environmental options needed in order to utilize the Core Gameplay. The Map gameplay provides the player routes to the other team's flag, it determines where the weapons are, what the floorplan is, it controls how the Core Gameplay is interpreted!

The Core Gameplay defines the player's toolset within the game, and the Map Gameplay gives the player a place where they can use the toolset to either their advantage or disadvantage.

Map Gameplay exists to augment and provide variety in the Core Gameplay. The term 'augment' is, perhaps, the most important aspect of Map gameplay. There are 10,000 + Unreal Tournament maps and even more Half-Life and Counterstrike levels out there. Use the Map Gameplay to give the player a reason to play level X instead of level Y. A map or level should offer more than the simple ability to run and shoot. All the other maps already offer that since those abilities are given by the game itself – the Core gameplay. The different ways a designer can enhance the Core Gameplay will determine if people will want to play the level or not and, ultimately, enjoy it.

The Core Gameplay is the foundation upon which the Map Gameplay is built. When beginning a game level, the house is not yet constructed. It is the designer's job to take the foundation and build it up until the house is complete. The Map Gameplay should add depth to the Core Gameplay.

The most common Map Gameplay mistake is to create a level that's simply too simple, and therefore adds nothing to the Core Gameplay. Almost every game out there that also provided a level editor has a 'box level' floating around on the internet. A 'box level' is usually a large cube where all the players or monsters spawn right next to each other and can do nothing other than attack each other. This kind of level adds nothing to the Core Gameplay. The only actions are the Core Gameplay actions themselves: spawn, run, shoot, die, rinse and repeat. The Map Gameplay itself is non-existent.

Some players do find such levels fun, but this is thanks to the game designer – not the level designer. The level designer essentially failed in doing level design – they failed to augment the Core Gameplay through the map's environment – the Map Gameplay.

Finding a game fun to play is quite different than finding a level fun to play. As a level designer, it is absolutely imperative to recognize and understand this difference.

'Good' Map Gameplay supports and motivates complex strategies and enhances the depth and fun already available through the use of dozens of elements. These can include traps, height differences (vertical or 'z-axis' gameplay), planned and balanced item placement, extra story elements, puzzles, and more. One example of leveraging Core Gameplay through Map Gameplay is the use of 'liftjumps' in the Unreal Tournament series. When a player gets on a lift, or elevator, by jumping immediately before the lift stops, they can propel themselves much higher in the air than normal. And depending on the speed of the lift, this can be quite high. Therefore, a level designer can help the player leverage the Core Gameplay by adding Map Gameplay possibilities around lift areas to reward the player for knowing how to 'liftjump'. For example, there could be an extra floor above the lift's normal exit that the player can access by 'liftjumping' and thus be rewarded by a Core Gameplay pickup. Thus, the level, the Map Gameplay, leverages the player's Core Gameplay 'liftjump' tool, by providing a reward for mastering the Core Gameplay. Without Map Gameplay, all the levels of a game would play the same. If a map does not augment the Core Gameplay, it fails in terms of gameplay.

Try to avoid focusing completely on either just gameplay or just graphics. Only a perfect combination and harmony between them can bring about a good game experience. Graphics can not only make the game attractive, but can also influence the gameplay, which itself supports the graphics. They interact and play off each other. A frightening monster setting needs specific environmental elements to function well, and vice-versa; specific environmental elements will convey the scariness of the monster. In a development world where gameplay is increasingly separated from the visual side of levels, I fear that this is often forgotten. Artists sometimes have little understanding of gameplay mechanics and gameplay designers often have little artistic insight. The end result of this separation can sometimes be that they both implement each of their visions into the environment and finish by working divergently rather than together. Don't make the same mistake – one cannot exist without the other.

FLOORPLANS

Although floorplans for SinglePlayer and MultiPlayer levels are quite different, both need to comply with some common basic rules. There are two general styles of floorplan: the Abstract, and the Realistic.

• The abstract floorplan

Abstract floorplans focus entirely on gameplay while the theme takes second seat. In an abstract floorplan, it's perfectly fine to place a hospital next to a factory as long as they look good standing side-by-side. It may not make a whole lot of sense, but it is fun. Many multiplayer floorplans are abstract; the levels in games like *Unreal Tournament* are perfect examples of this. Games like *Quake* also make extensive use of abstract floorplans, sometimes even in singleplayer levels. Other games like *Tomb Raider* and *Serious Sam* also use abstract floorplans.

• The realistic floorplan

Realistic floorplans meld the theme and gameplay together when constructing the floorplan. For example, in a factory, many different areas are needed: storage areas, transportation areas, fabrication floors, control stations, and more. Most of these rooms and areas would need to be laid out in a logical order following the process of whatever the factory makes. For example, control stations would accompany each fabrication area. Then storage areas would be placed ahead of the transportation areas because the items are usually stored before being shipped out. Perhaps the transportation area is both the beginning and the end of the cycle, in which case it needs to be at the head of the area chain, and the back, which would create a circular floorplan. The layout should make 'realistic' sense and serve to accomplish more than simply support gameplay.

Games that place a heavy emphasis on story are of laid out in this manner, for example *Morrowind* and *The Darkness*.

It is important to understand that both areas share pros and cons.

An abstract floorplan is usually much easier and faster to make. Therefore, if time is a limiting factor in the design, then picking an abstract floorplan might be the best choice. In addition, because they're abstract, they allow the designer to put more focus on the gameplay; especially in games with strong gameplay and weak stories.

On the other hand, realistic floorplans are more difficult to implement correctly, but will look better if done right. If the game also has a strong story and theme, then a realistic floorplan may almost be a necessity. Games with strong stories always have realistic floorplans. This is also true for games that rely heavily on certain well known themes/environments. If the level takes place in a famous city, like London, the floorplan should take into account the general layout of the city. Conversely, a level that takes place on an alien planet would do fine with an abstract floorplan.

Realistic floorplans also tend to be easier to understand. Casual players are usually more interested in levels that seem to resemble something in the real world than levels that are hard to decipher if the theme is too strange or foreign. If the level's, or game's, target audience is casual, it may be a good idea to follow a realistic floorplan.

Of special note is *Portal*. The test chambers in *Portal* are very abstract in such a way that they do not resemble anything specific. They exist solely for the gameplay. This is perfectly excused, however, by the fact that they are in a test facility, which means that they don't have to adhere to anything other than the test facility. The game manages to portray its levels in a realistic way while at the same time, the levels are fairly abstract. This could also be an option, but it must remain carefully balanced.

Of course, in the end, the choice is up to the designer. Just be consistent and stick to the chosen type. If the two are to be mixed up, blend them together and balance them well so that they're not obvious. I recall a very vague area in *Resistance: Fall of Man*. The entire game had more or less realistic floorplans, yet in one area the player entered a small room through a doorway blasted through a wall. The small room contained a huge door and a plaque next to it with directions on it. Why would a room with only one, very obvious, entrance need to display directions on how to get out? Why is there a huge door for a little room? It just does not make sense. Such situations break the immersion and remind the player that they are just playing a game.

Regardless of the chosen floorplan, the player should never get lost. Being lost is frustrating and is not a valid type of gameplay. Make the player's path through an area very clear, or at least clearly demonstrate where they can and cannot go. Pathing hints can be almost anything: colors, tones, enemies, items, audio, and more. Many games have monochromatic color schemes and are therefore difficult to navigate through because the whole level looks identical. The previous example, *Resistance: Fall of Man*, suffered from this. Bioshock certainly did not as there was often plenty of contrast to help the player navigate. Throughout the book, this concept will often be revisited.

MULTIPLAYER

BASICS

Gameplay, especially in fast-paced multiplayer-oriented titles like Unreal Tournament and Quake, are often misunderstood. Most peoples' opinion comes from watching over the shoulder of someone playing the game. Therefore they usually come to the mistaken conclusion that the primary goal is to create as much blood and body parts as possible. This is certainly not the case.

Talented players who leverage the gameplay understand that the gameplay is more akin to a 3D chess games clothed in fast movement and over-the-top weapons. The level, ideally, should support and enhance this – which is exactly the goal in multiplayer levels. Far too many people create over-simplified floorplans and never touch any of the finer points of gameplay. Why would a player choose this level over that level? What does this level offer that that one does not? How does this level add more depth to the game? How does this level promote teamwork, and that level not?

FLOORPLANS

So how do floorplans fit into the grand scheme of things?

A floorplan should provide flow and the ability to use the area against the enemy. In other words, it should offer strategic potential to the players in the space. The level should offer the player opportunities to think. A few simple rooms connected by corridors is, in most cases, simply not enough because it adds nothing to the core gameplay. Ideally, it will offer extra depth and potential.

The floorplan should encourage the player to move around easily, otherwise known as the 'flow' of the level. Rooms with only one exit/entrance, long hallways, and dead-ends are all level designs that interrupt and stop the flow of a level. Design the area to promote speed, possibilities, and depth. All of these will be addressed in more detail later.

If the player must pause to ask themselves how to return to a previous area, then the flow is lacking. A player should be able to get anywhere in the level from any other location in it without too much thinking. The connectivity should be intuitive – so much so that it almost pulls the player through it all by itself. The player should be able to see the entire level by just constantly moving roughly forward.

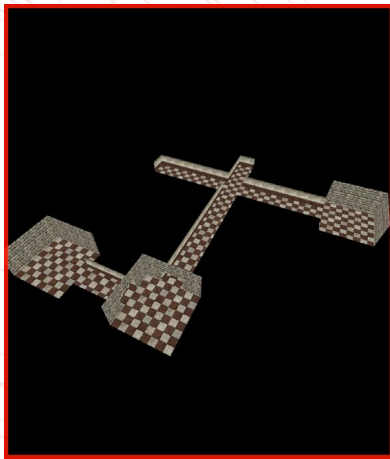


Image GP 1

This is a bad example: Long corridors and several dead ends. These both remove the choice of where to go and what to do. There is only one way in and out: it is a trap.

A player who goes in will be trapped once an enemy appears. There's little to no reason for a player to enter such a dangerous area. This can be a way to promote a risk/reward situation for the player, but the reward would need to be fairly large for such a large risk.

The following example is an improved version, but it still isn't perfect.

There may be two entrances/exits, but this is still insufficient. How can a designer offer choices and strategy to the player when they are presented with only two options, which quickly becomes just one option if an enemy enters the room. Thus, the player's behavior is predictable – they will likely attempt to leave through the remaining doorway. There's no strategy in this situation.

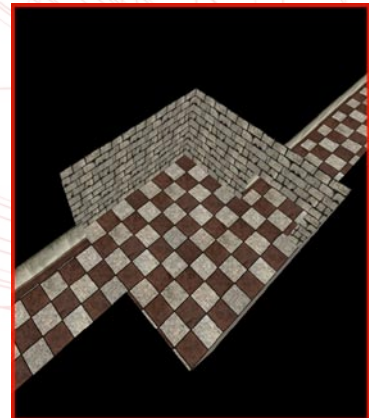


Image GP 2



Image GP 3

The best situation is to have three or four exits, as can be seen in the following example.

Why is this better? It's better because there are multiple paths in and out. It flows; the players are given enough opportunities to allow them to make their own decisions instead of being forced into a single corridor. This opens the way for a more variation in the play, and more strategy.

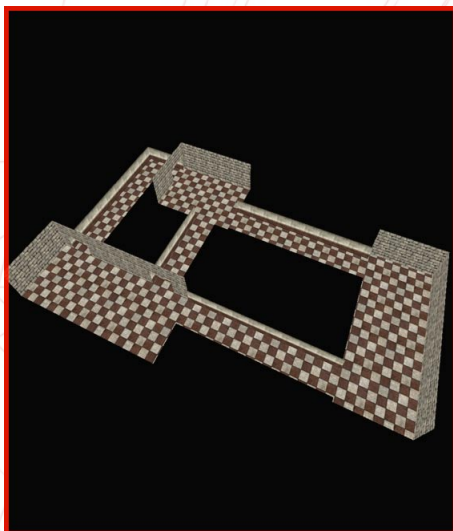


Image GP 4

Connections themselves, for example corridors, can lead to other problems.

In this example the corridors are entirely too long which is not ideal.

The result is a three-fold problem. First, the gameplay becomes overly linear and straight. Second, there's little gameplay in running down straight long spaces. And third, the player is trapped which results in the same problem as the room with two doors – it's too predictable. The player only has one way to move when an enemy shows up on one side and the long corridor makes aiming easy. This endless corridor situation is often described as a Room-Corridor-Room, or RCR, situation. Unless the core gameplay calls for it, this RCR scenario should be avoided.

Corridors should be short, or a medium length in general. Connectivity can also be added by intersecting the main corridor with a side corridor, or even by adding a hole in the floor leading to a different area. The following screenshot shows how a long corridor can be intersected by two other corridors and have a turn in the middle to give the player more freedom of movement, and block the view a little.



Image GP 5

A long corridor is not always a bad thing though. It can work well in singleplayer games or in specific multiplayer scenarios. In game modes like Deathmatch or Capture the Flag it could be exploited as a danger zone. This will be covered a bit more later on.



Image GP 6

Another common mistake is creating areas that are too large and/or open. Exceptions to this rule largely depend on the core gameplay. Examples of this are games that factor in mixes of infantry and vehicles, for instance the *Battlefield* series. In most shooters that do not involve vehicles, like *Half-Life*, *Quake*, or *Unreal Tournament*, these areas are to be avoided, like in following good example.

In this example the large open space is successfully broken down into smaller areas by adding two large walls in the middle, and several levels and platforms.

The issue with too open areas is that they rely too heavily on Core Gameplay and aiming. Since the enemy can always see the player, the player's behavior becomes very easy to predict.

Open areas > the player can only jump around and attack > repetitive > predictable > boring

The gameplay goal in these types of games is more than simple aiming skills and some Core Gameplay. Areas should be broken up, closed, and opened in interesting ways to fully enhance the map gameplay beyond the Core Gameplay.

As previously mentioned in the Design Chapter, an important, but oft-neglected aspect of floorplan design is supporting the visual and technical requirements of the game. When I design a floorplan I keep the visuals and technical requirements in mind at all times.

Both of these elements are very important when designing floorplans. Certain rooms, such as large open cubes with catwalks are more difficult for an artist to decorate which increase demand on time and effort. Why not make it easier? A well-designed floorplan will not only help gameplay, but also the visuals and the framerate. One possible issue that can crop up with areas that are too open is that the area may end up being too difficult to optimize with occluders.

The point is that some shapes and sizes can make a level easier, or harder, for an artist to turn into something nice and well-optimized. If the designer is not responsible for the decoration, then talk to the artists who are. If the designer is responsible, then try to imagine how the visuals will flesh out the floorplan ahead of time. How will this room be populated? What kind of architecture and lighting will go well in it? What changes could be made to make it easier to decorate? Would it look better if the ceiling were higher? Perhaps all the interesting angles actually can't be filled with modular assets.

The level, as a whole, must remain in harmony with all its disparate components. Levels are populated by different assets and actors that are created to work together – not independently. Therefore, even when one's task is to only create floorplans, it's still necessary to have some sort of knowledge about all areas of level design. This is another reason why I, personally, am not fond of the trend of splitting up level design into many different design components as actual positions in a team.

ITEMS AND POWER-UPS

One very important aspect of multiplayer games is item and weapon placement. The pickups will determine where the player will travel through the level and the subsequent routes they choose. Pickups create goals and destinations and thus ambushes as well. Pickups can help the player by providing them with health, ammunition, and extra tools, but can also work against them by giving away their location and therefore endanger them. Usually, when a player picks up an item a sound is emitted. If an enemy is near enough to hear that sound, then they will figure out where the player is without having to see them. The level design should grant the player these risk/reward choices and facilitate the consequences of the player choosing to pick up the item, or to pass it by.

This can take the form of many small pickups with meager benefit, such as small armor bits, ammunition, small amounts of mana or health, or larger health portions at key points or connections in a level. They can be placed in doorways, corridors, on stairs or ramps, but with enough room for players to pass them by if they so choose. Give them the choice to gain some benefit but reveal their location, or to pass by silently but miss a slight boost. Powerful pickups, on the contrary, should be located in riskier locales, for example where the enemy can easily ambush the player, in order to offset the increased benefit the powerup gives.

The dead end in this screenshot contains a pickup that doubles the weapon damage. However, if the player decides to pick it up, they will cause a sound to be emitted, and nearby enemies will be alerted. They can then rush the player in the dead end. It's a dilemma. The benefit of being stronger is offset by the risk of imminent attack and no easy way to escape. This therefore balances the gameplay. The player should not get powerful items for free – make the player work for them. The more powerful or special an item is, the more the player should have to work to attain it.



Image GP 7

Another example of a simple floorplan trap is a corridor shaped like a donut. With the entrance and the exit right next to each other, any player entering the donut becomes trapped. Compared to a straight dead end corridor it does give a player a few extra options, but it still is dangerous.

Other examples would be to place the pickup in, or on, a trap like a floor that can collapse or in water. In most games, water slows down players and players moving through water make noise. Thus it's dangerous because the player constantly makes noise and cannot move as fast. In other situations, sometimes a door is placed in front of the pickup to make sound, or the pickup could be at the top of a tower where all the enemies can see the player taking the pickup; also the player picking up the tower item could be at risk of a long fall.

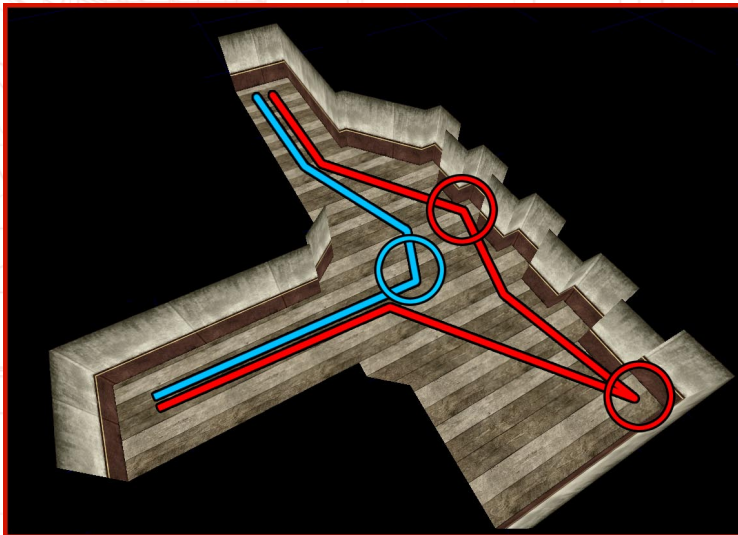


Image GP 8

Items determine how players travel through a level therefore they should be easy to get to.

The red line illustrates bad item placement and the blue line a preferred placement. Following the red lines, it should be clear that the extra moves used to grab a few simple items in the room shouldn't be needed. Allow the players to spend their time on more important gameplay instead of on collecting pickups.

Because of the above, I am not a big fan of games or mods that contain no pickups. Well known examples are Counterstrike and Unreal Tournament's Instagib. By taking away items, an entire layer of gameplay depth is wiped from the game. If it is impossible to add items to the level, then attempt to compensate them through environmental substitutes, which is never a bad idea, even when items are present. Environmental substitutes could be puddles

of water that give away player location based on splashes. Since pickups usually also emit sounds, puddles in their stead would work as well. Another common example would be an old plank emitting a creak when the player walks over it. More examples include shards of broken glass, doors, a movement-sensor attached to an alarm, and more. Not only do these substitutes enhance the gameplay, but also the theme and audio.

TRAPS

Traps can work in both Single and Multiplayer gameplay and can add fun, or frustration, to the game. They are especially liked by more casual players. However, just as in singleplayer games, keep it fair. Players should recognize the danger as soon as they see the trap – before they actually encounter it. An unfair example would be a ‘secret’ trap that kills the player with no warning. The player should be in control of their own fate in the game, or at least have that illusion. A good example would be the trap from Unreal Tournament’s DM-Pressure level. A special chamber in the map holds a few powerful items, but the chamber can be closed and pressurized by players outside the chamber. The players in the map clearly see, and interpret, the danger and the designer offers them a distinct choice: pass it by and live, or enter the chamber to grab the items, but take the risk of getting killed?

Designers often add doors that open and close to levels. While they can work well to block a long sightline or to split up an area, they can also become inadvertent traps. The door should open fast enough that the game flow is not obstructed unless it’s a special door. If the player is in a fast moving action game, then waiting for a door to open is not an option. Likewise, make sure the door closes only when there’s no chance that a player can get caught in it. Being killed by a door in a game is very frustrating. Make sure the player spends their gameplay time on gameplay that matters and avoid causing ‘accidental’ deaths. However, doors can also be used for traps. What I occasionally do is add a door that is normally open, but can be closed at a push of a button. The door remains closed for just a few seconds, but this gives the player an opportunity to temporarily trap or delay the other player. It introduces a small bit of extra strategy to the map gameplay.

Another possibility is adding a door that opens really slowly, but once open, it offers a short cut to another section for just a few seconds, before it closes again.

PHYSICS

In recent years physics have begun to open more gameplay possibilities than have ever been available before. One of the more popular uses has been to use pieces of the environment as weapons, for example *Half-Life 2*, which has made great use of that. Imagine the fun of defeating your opponent with somewhat ridiculous objects, like a refrigerator.

If the game supports an adequate level of physics interaction, then make sure that furniture or other movable objects don’t interrupt the flow of the level, or give players opportunities to abuse the system to unfair benefit, most especially in multiplayer games. Imagine how a player with low health would feel if they were running away from an opponent and they suddenly find their escape route blocked by furniture one of their teammates bumped into and moved to the doorway. This could be very frustrating.

Or, instead, perhaps the game is objective-based, like Capture the Flag, and the opposing team has taken all the physics objects and piled them on top of their own flag so that it’s now unreachable. Perhaps it would be funny the first or second time, but eventually it just becomes an unfair tactic created from unintended core gameplay.

Such situations should be avoided entirely and it is the designer's job to ensure core gameplay consistency across the map. A possible solution for the above dilemma could be to respawn the objects back in their original locations a set amount of time after they've been moved.

Especially true in multiplayer, ensure physics cannot affect the core gameplay – only level gameplay. An example of a 'bad' physics puzzle in a fast-paced shooter would be that the player needs to collect a few wooden planks before being able to reach the flag. The core gameplay is all about movement and action, yet the task of rearranging wooden planks takes time and slows down the player – thus invalidating the core gameplay – and the player is asked to do this very close to the objective which is when they'll likely be under the most amount of fire. A fair alternative would be to use the wooden planks far away from the flag, perhaps as a means to create a short-cut to the flag area. Although the situation still grinds a bit against the core gameplay, the situation is much more fair to the player as they're much less likely to be under attack when they attempt it. It is an extra step that is not necessary but may pay off and help the player if they decide to put in the time and effort. Make them optional extras instead of necessary tasks.

COVER

If the game uses a cover system, and this goes for both single and multiplayer, make sure that there is no perfect cover spot. Manipulate the cover and the geometry such that every single piece of cover available has a weak angle from which it can be easily countered - either from the left or the right, front or back, or perhaps even from above or below.

Direct gunfire does not have to be the only way the cover can be invalidated. Some kind of trap could be placed near the cover that can be activated from afar – by a button behind enemy lines, for example. Another example could be a piece of cover that is obviously well protected – more so than the surrounding cover – but then add a button that lowers it into the floor, or opens a trap door underneath it. The button should be quite visible and within range of the cover spot so that players that are in cover would have the opportunity to fire at enemies moving toward the button, or move away of the cover spot. Remember to let the player decide their own fate.

Countering camper behavior is a beneficial side-effect of avoiding 'perfect' cover spots. Try to ensure that the cover is designed in such a way that campers cannot find areas with strong protection that also shares a long overview of the level, and has health or ammo. Add a second entrance to the cover area that's easily accessible – a long walk up four flights of stairs does not qualify. It could include super-fast methods such as using elevators or jump pads – anything to allow other players quick and especially fair access to a counter-attack.

AURAL FIXATION

Sound is an oft misused but extremely important aspect of gameplay. It can give away the player's position, or help identify where an enemy is, and also aids in the player's immersion in the world. The level should support and reflect all of these.

For instance, if more than one elevator is present, then giving each a different sound, even by using different pitches, helps players localize the source of the sound. If every elevator sounds the same, then the player would not know which elevator the enemy is using; the sound could be from any one of the elevators.

This is the same reason why to place environmental sound emitters, for example puddles of water or rusty metal pieces. Whenever players travel over them, the sound emitted gives away their position. This is also the reason why designers often place small pools of water in corridors or other areas – walking through them makes a sound and thus acts almost like an alarm for the other players.

Also, each item and pickup in the level should have its own unique sound. Thus the player can clearly hear what the opponent has picked up and where they are. High-pitched sounds are easier to hear so they should be reserved for significant powerups.

Other level pieces that can be used as ‘alarm’ systems for other players include doors, jumppads, teleporters, metal detectors, and more. Ideally, each should use a different sound to clearly give away their location.

A level in which sounds were used with great success is CTF-Coret from Unreal Tournament. Each flag room had two exits – one with a jumppad and one with a door. Whichever exit was taken, the defenders knew which route the enemy took because of the associated sounds. A smart enemy could counter this by first activating the door and then running back to use the jumppad. This is exactly the kind of extra gameplay depth that sound can add to a level. We will return to this example shortly.

IN-DEPTH EXAMPLES

CTF-CORET

I personally love CTF-Coret, the Capture the Flag level from the original Unreal Tournament demo. Its floorplan is relatively simple in that it only has two primary routes to reach the other base and, despite the lack of large open areas, it works well.

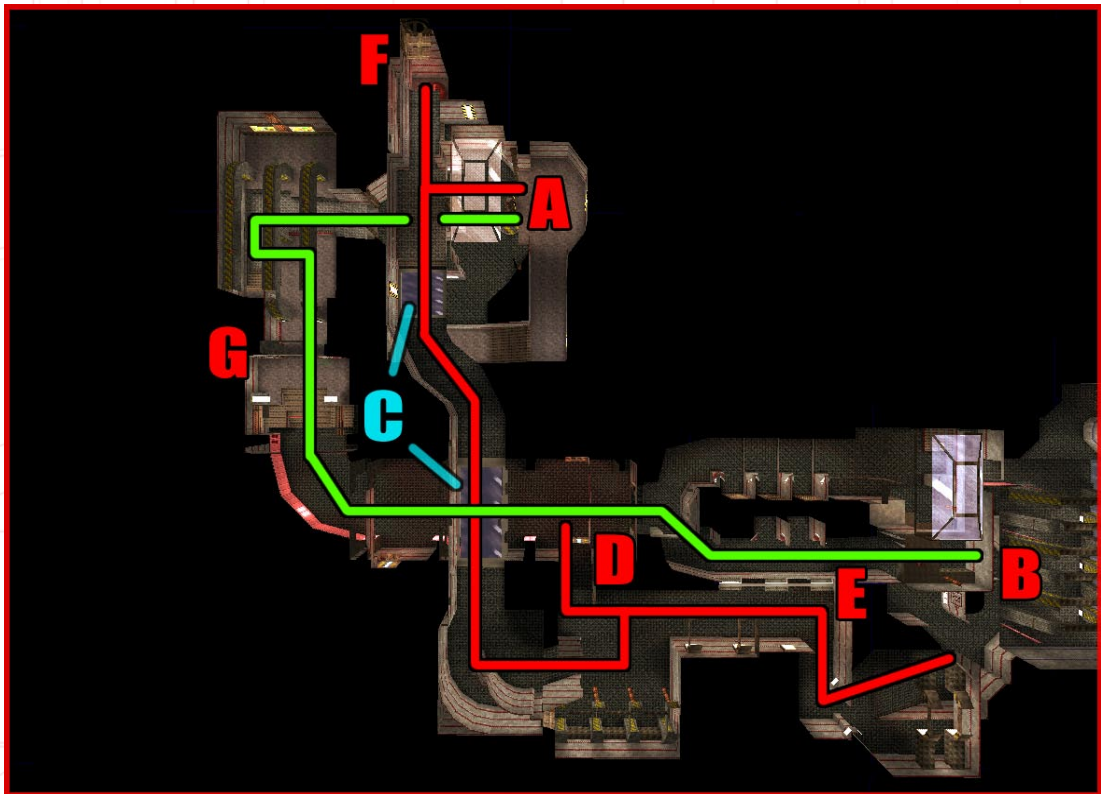


Image GP 9

The flag (A) can be reached via two routes: the upper, red route, and the lower, green route. The upper route can be reached by using a jump pad in the flag room (F), while the lower route will guide the player through a large door (G). There are two glass floors between the upper and the lower route (C) which allows players to see into the other route. Half-way down the middle, the routes cross and intersect several times which allows players to switch routes (D). The wall in-between the two routes contains a hole through which players can see and shoot at players using the other route (E). A third route also branches off the lower route half-way across the level, and continues to the midfield (B) following a slightly different path.

Now, how does all of this impact the gameplay?

Both the hole in the wall (E) and the two glass floors (C) temporarily reveal the position of the enemy to the player, and vice-versa, thus forcing each party to try to trick the other, or else the enemy will know where they will end up. The player could, for example, simply pause for a bit to mess up the other player's timing, or they could return and take another route instead. When the enemy has stolen the flag and is trying to return to their base it can also reveal the position of the enemy flag carrier.

In the flag room, the door on the lower route (D) and the jump pad to the higher route (F) follow the same logic. When a player uses the jump pad, or travels through the door, each makes a distinct sound, thus alerting other players. Choosing either route is not without risk as it gives away the player's position.

The floorplan only has two main routes which is usually ideal for a CTF map. Too many routes can make the level too difficult to defend since the enemy could come from almost any direction. Also an enemy flag carrier could switch paths too easily to mount a defense against. The focus in team-based game types should be the struggle between the two teams – not on which team can play hide and seek the best. Don't give too many options, and keep them focused. Ensure that there are enough central areas to reconnect routes. Usually there is one in each base area and another in the midfield although there can be a few smaller ones along each route, as in CTF-Coret (D). This idea can also be seen in the floorplan from CTF-Lopo which we'll examine next.

The players need the opportunity to change routes along each path so that they have options while trying to avoid the team chasing them. Without this ability, the design would have the same problems of the corridor example discussed earlier where the player's options are too limited to provide meaningful gameplay.

Most team-based levels work well with focused areas in middle of the map and in and/or around the flag base or end objective area. This usually is implemented by constructing fairly open base areas with multiple routes in and out. Initially those routes out of the base are fairly limiting and pretty safe, but soon open up more either in terms of space, or in terms of interconnecting paths. Once in the midfield all the routes connect again. Basically the floorplan boils down to balancing open areas against chokepoints throughout the level. Players can lose their pursuers temporarily while leaving the base but they may be faced with them again in the midfield or in other chokepoints, which balances the gameplay again, ensuring that there is no perfect route which is one hundred percent safe...

CTF-LOPO

CTF-Lopo is a level I made for UT3. The focus of the mappack in which it was included was on great gameplay in a low-polygon environment. It has a more complex layout than Coret, but follows the same core ideas. One difference is that Lopo adds more variation to the gameplay both in terms of height and in the type of combat.

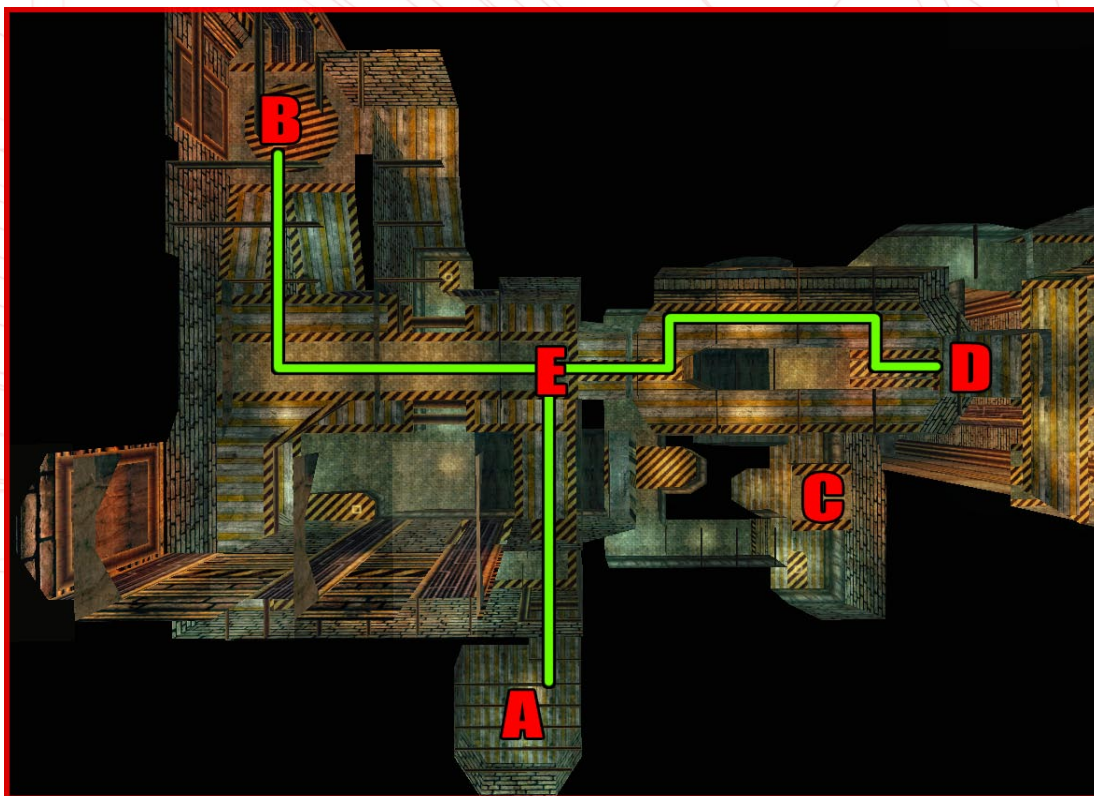


Image GP 10

The flag is located within a side room off the team's base area (B). Most players start off in a spawn room (A). The level consists of two primary routes; an upper route (green) and a lower route (see the next screenshot). The midfield area (D) is a large open area (not pictured). Players have the opportunity to switch routes half-way through the level (C).

Similar to CTF-Coret the midpoint of the map allows the player to travel from the upper route to the lower, or vice versa. It's somewhat small and simple (C) – a narrow corridor with a jumppad to propel players to the higher level. Yet here, the sound the jumppad makes when used can indicate to the other players that the player is changing routes from lower to upper. There is also an armor pickup in the area and is located slightly separated from the main routes so that players will need to invest some additional time and effort in order to pick it up. If it had been placed in the middle of the main route there would be too little risk and all reward for the players.

Although there are only two primary routes into the bases, there are many more routes available once inside. This also occurs in the middle. The area in front of the base acts as a chokepoint but connects to more open or varied routes. This encourages the defending team to plan a defense: do they want to defend close to the flag but potentially be caught by surprise due to the large number of routes immediately around the flag? Or do they want to defend the chokepoint where it may be easier to repel enemies, but leaves the flag itself somewhat unguarded?

The spawn area (A) is in a 'safe' corner where players can spawn and easily get back into the game through a number of routes through the base (E). This allows the defending team to either rush to the flag area or the midfield fairly quickly. If the spawn room was all the way in the back of the base, for example in the bottom left, it would be very difficult for newly spawned players to catch up with the flag carrier as there would be too much space to cover.

The upper route itself has some variation within in: it splits into two routes – one wider than the other – for a few meters. This adds some complexity to the gameplay as both require slightly different tactics. The midfield carries this idea one step further with a very open area. Most of the level focuses on close combat and indoor fighting, however the midfield area switches this up and presents the players with a wide open outdoor conflict area thus adding additional variation to the map gameplay.

In the base area there is a ramp that allows players to travel from the ground floor (A) to the next level up (B) and is bordered by a wall with holes in it. If a player sees an enemy taking the ramp up, rather than chase them up, the player can stay on the ground floor and fire at the enemy through the holes in the wall. As soon as the enemy ascends the ramp, their position is revealed which puts them at a disadvantage. The holes in the wall (C) help expose them and they're also fun to try to shoot through which adds to the satisfaction of having killed an enemy through them.

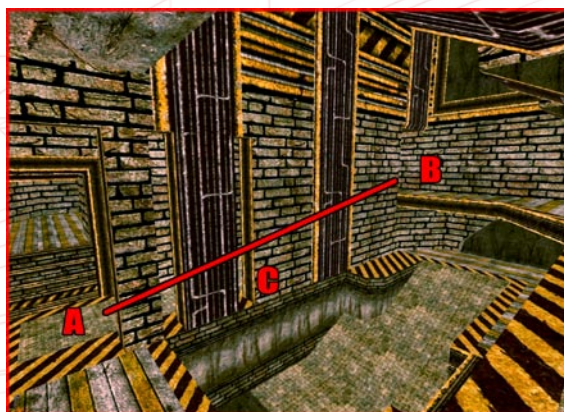


Image GP 11



Image GP 12

Now, let's examine the lower route. The base area is to the left and the right leads to the midfield. The routes on the right side split – just as in CTF-Coret. Once past the chokepoints, in the midfield, the routes combine again. The route from the base to the midfield is split into the primary, red, route, and the slower secondary green route. The green route is narrower and thus more dangerous as there is less room to avoid gunfire and therefore usually goes unused. The main route

has two large shallow pools of water (A) which give away the position of players traveling through them. And so the players are faced with a dilemma: give away their position but stay safer and use the shorter route of the two, or travel without noise indicators, but spend extra time in a more dangerous place. Regardless of which path they choose they will have to traverse one pool unless they translocate up (a teleport device in the game) near the left pool. However, translocating is likely to reveal their position anyways since there are sounds and particles associated with teleporting. Therefore there is another dilemma.

FINAL THOUGHTS

When designing a level with the player in mind, always think about what the player sees their first time playing the level. If someone playing the level for the first time cannot find important elements without purposefully searching for them then the design is flawed (unless it's a secret). All the fundamental items required for play should be clear on a player's first time in the gameplay area. Traps should be obvious and objectives must be easily discovered. If a level does not offer a certain level of accessibility, or 'ease of use', then people may just quit and play something else.

Certain elements can be added with more experienced players in mind and this is actually encouraged as it increases the depth and life of a level, and possibly the game as well. But a new player should be able to find the weapons, objectives, other basic items, and not get killed by sudden invisible traps. Frustrating the player does not promote re-playability. Keep it simple for beginners, but provide enough more complex aspects to keep more experienced players coming back for more.

Core gameplay and map gameplay are two separate aspects but map gameplay does have its foundation on the core gameplay. For multiplayer, this means offering enough variation on the core gameplay, without altering that core gameplay. Allow the player to add their own tactics to the game. Design levels with enough tactical potential that players can experiment with strategic choices. Satisfy the beginners who will enjoy pushing a button that causes a giant crushing object to kill an opponent, and also the advanced players who enjoy complex floorplans that allow them to mislead and predict their enemies' locations. Beginners are easily pleased by simpler, more obvious level features, while the experienced players will squeeze every ounce of gameplay out of a level. Strive to give both what they want without compromising the other.

SINGLEPLAYER

THE BASICS

The core of singleplayer gameplay is the story, the player's abilities in the world, and the interaction between the two in the level. The over-arching story spans, usually, the entire game with secondary events and/or themes that persist through a few levels. Because the story is a trait of the game as a whole, it is therefore part of the core gameplay experience. The level should not exist outside the storyline in the same way a player should not be able to modify the jump height or walk speed as defined by the game's programming. It can elevate the story and the core gameplay to a higher level, in short, enhance them.

A significant portion of map gameplay in singleplayer is communicating the story to the player in interesting ways. Instead of simply displaying the story through text, the player should experience it first-hand. Show the story to the players and get them involved in it. Try to avoid the player feeling like they are simply visitors to the world, or they are looking through a window (or monitor) at it. The created world must be interesting and compelling to play and experience. Keep it interesting by combining visual and gameplay elements in the level.

On the visual side, levels should contain enough thematic differences to prevent them from becoming repetitive. I personally found games like *Doom 3* and *Bioshock* to contain too much repetition because most of the environments throughout each game gave me the impression of being too similar. By being so repetitive, the levels weakened the atmosphere, story, and immersion of the player.

In addition to variation, the levels should hold enough style, atmosphere, and significant events to support the story. Suck the players in and make them feel like the world around them is real. Atmosphere is needed to make the story believable. Atmosphere is related to emotion and, some exceptions aside (*Serious Sam* & similar), most game stories invoke emotion. The levels should reflect the emotions through atmosphere and events.

On the gameplay side, players can be dragged in through AI behavior, events, unexpected story twists, the environment, and, again, atmosphere. Many designers make the mistake of only examining the gameplay side of the story. Try to avoid such a narrow focus. Singleplayer gameplay draws heavily upon audio and visual cues and vice-versa. It all must balance evenly in order to work well together – it cannot work together when separated. Setting up a scary monster encounter will be less impactful if the environment does not match the scary mood of the dangerous situation. Likewise, the frightening environment will be less so if the enemies do not scare the player through their placement and behavior. The connection between the audiovisuals and the gameplay is much stronger, usually, in singleplayer levels rather than in multiplayer.

This chapter will not cover AI behavior specifically because I regard it as part of the core gameplay. The focus is on map gameplay and not on the ways enemies attack, the behavior of weapons, their reload rates, etc.

Let's move onto a number of important elements of singleplayer levels, and speak about them more in depth.

AI PLACEMENT AND BEHAVIOR

Many people make some common mistakes when dealing with AI placement. Often, AI are placed without giving them a task, or without 'hiding' them. In many games and mods, it's easy to find an enemy standing still, simply waiting for the player to wander by. This usually feels quite unnatural. Enemies are supposed to be part of a living, breathing, game world – it's the place they live and work. This should mean they have their own 'job' in the world. Put them to work! If it looks like enemies are just waiting for the player, it ruins not only the credibility and depth of the level, but also destroys a player's suspension of disbelief. The player should feel like they're in a living world that exists on its own. Don't make them feel like they're the center of it and it's made just for them, otherwise it becomes clear that the world is just an illusion. Have the enemies and/or NPC's perform tasks in the world: operate machines, clean weapons, make fires, work on computers, patrol, have conversations, prepare food, read newspapers or books, hunt for food, protect their young – just about anything other than standing still.

An enemy that is guarding something, for example, does not simply stand still as in the screenshot on the right (yellow circle).

For a player, it would be awkward to enter the area (red arrow) and be faced with an enemy standing still, looking straight ahead until they walk by. Perhaps the enemy has recently had some head trauma, or that it considers staring straight ahead a jolly good time, but to the player it will simply not look natural. Again, put the AI to work – give them a purpose in the world.

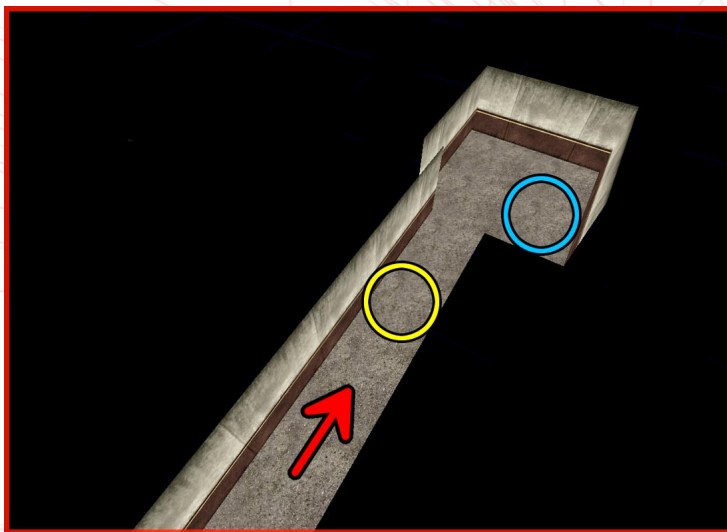


Image GP 13

If setting up their individual jobs is beyond the scope of the tools, or too time consuming, then at least hide them from the player until they're 'activated'. It's alright to have enemies wait for the player as long as the player doesn't see them waiting. In this example, the enemy could be waiting around the corner, in the next room (blue circle). When the player hits a trigger in the corridor, the enemy 'hears' the player, and reacts by running towards the player. This is much better than simply having the enemy obviously waiting in full view of the player. The NPCs should add depth and credibility to the level through their behavior. Standing still in full view easily breaks the illusion of a living world.

Also, be sure to incorporate variation into the NPC placement. Spawning the enemy behind a door every single time eventually becomes predictable and boring. I found the first few levels of *Doom 3* to be rather scary because of the sudden way monsters spawned near the player. However, after about an hour of play, I managed to figure out where the next monsters were going to come from after entering a new area, before there was any sign of them. The game maintained the same approach of spawning enemies throughout the game which softened, or deadened the experience and thus ruined the scary atmosphere. At the same time, it might be wise to repeat some of the NPC behaviors. By introducing some predictability into enemy behavior, it can help the player feel more powerful, or feel like they've 'outsmarted' the NPC's. This can also help the designer in terms of pacing a player's experience through a level. It may be that the designer allows the player to get used to a

particular NPC behavior by using it multiple times, or in certain situations. But then the designer can switch it up and slow the player down (or speed them up) by introducing or re-introducing another type. Just about when the player feels they understand the system, switch it up and replace it with something else. Variation, like most aspects of level design, needs balance. Too much will ruin the experience, too little could cause the player too much frustration.

The type of NPC is also important. Here, again, bad choices can ruin the level's credibility. NPC's should fit the environment they are placed in, according to the overall theme of the game. Generally, medieval knights should not be placed in a futuristic level. While this may seem obvious, this mistake does get made. Creatures or people living in a certain environment should reflect the environment. The environment influences them, and they influence the environment. Make the connection between the two clear. Whatever lives in the game environment should fit the theme in terms of behavior, art style, color, etc. Several types of mismatches can exist: style mismatch, theme mismatch, and color mismatch. An example of theme mismatch would be the knight example mentioned above. Style and color mismatches are often even more important than theme mismatch. Depending on the game's story, or the particular circumstance, maybe a knight does exist in a futuristic world. If so, the knight should still match in terms of style and color even if there is no theme mismatch. It is more important that objects in the world look and seem possible than if they are really logical since people often make judgments at first sight

The opposite can also be true. Sometimes thematic NPCs don't blend in. An example may be a swamp beast that looks out of place in a swamp because its colors do not match the swamp's green and wet look. In this case the type of NPC would fit the environment, but it would still feel out of place and wrong because of its color.

An example of bad styling could be a photorealistic soldier in a very cartoony environment, or a very clean character in a dirty world.

In addition to the above, it is also important that the characters fit into the surrounding environment and that the number of different characters is restrained to a certain 'palette' in order to remain consistent. Forty different characters in the same environment would probably introduce some inconsistencies into the game. Colors, styles, and themes – they must reflect the level and remain consistent.

The inverse holds true as well: if all the characters in an environment are of the same type, the player would quickly become bored. Here, too, the balance is the important aspect. Variation is needed, but it must remain balanced. Stick to a moderately sized, yet varied set of characters whose various skills and attack methods complement each other. One enemy that prefers close combat, and one enemy that prefers gunfights for example.

Invincible and/or invisible enemies are almost never a good idea. Just as with traps, enemies should be fair and reasonable. Anything that looks impossible, or is impossible, will frustrate the player at the risk of shutting off the game. Enemies that damage the player yet can't be seen are not fair to the player. The same is true for the typical boss encounter where the boss will not die, or sometimes won't take damage, until some hidden button or trigger elsewhere in the level is triggered.

To avoid seemingly impossible situations give the player clear signals such as helping them identify who, what, and/or where. A sniper can sometimes be an invisible enemy, or a trap, depending on how it is placed in a level. The player should be warned that there is a sniper before being killed or damaged by it. This could be done by having a sniper shoot the ground in front of the player, or by having friendly NPCs near the player get shot. Just about anything would be better than letting the player die without warning. In *Half Life 2* the snipers had a blue laser that showed the player where the sniper was located, and what they were aiming for. If the player ends up having to use a save game slot to find out there is

danger, then that's bad gameplay. Gameplay should not be based on saved games – they are not part of core gameplay systems. Punishing the player by death should almost always be avoided – if the player dies, they should relate their specific actions and mistakes to the cause of death – not simply the game killing them on a designer's whim. Danger should add to the experience and tension in a level – not detract from it through frustration.

The player should also be able to clearly identify where the sniper is – where the danger is coming from. Signals such as muzzle flashes, or light glinting off the scope, will balance the situation and give the player some measure of control over the experience. Place the gameplay emphasis on the challenge of the fight – not on a scavenger hunt for a too-well hidden sniper. If the player enjoys finding needles in haystacks then they probably would have purchased a different game. Most action/FPS gamers do not usually enjoy that kind of gameplay.

This also goes for invincible enemies. A very strong boss enemy who is invincible at first glance should communicate hints to the player as to how to approach the situation. And when the player does damage the boss, there should be some feedback to the player that they've done the right action. This is why many boss fights involve pain animations and/or pieces of armor or body parts falling off the boss when they get hurt. For example, the titans from Unreal played pain animations when they reached low health. Another example of a boss status change would be when they change how they attack the player, or they run to another part of the level. Basically, some change that indicates the player is making progress.

If the enemy can only be killed by exploiting a weak spot, or through the environment, this should be communicated to the player as well. Clearly illustrate what must be done and what to look for in order to do it. One example would be to switch to an in game cutscene that introduces the boss, but also shows something like weak pillars. The player may get the hint that if the enemy goes under the pillars, perhaps the player can damage them and cause some sort of collapse on top of the enemy. Another approach could be having an NPC nearby that tells the player about a particular weak spot on the enemy. Regardless of how, the key is communicating information to the player.

SCRIPTED EVENTS AND GAMEPLAY VARIATION

One major element of singleplayer games are scripted events. What are they exactly? Scripted events are singular situations in a game that usually happen unexpectedly and help reduce gameplay stagnation, which is something some designers seem to forget. I have played countless games where there was seldom any variation in the core run/shoot/jump gameplay. Without scripted events, these games ended up feeling repetitive and, eventually, predictable. Always provide variation to entice the player to keep playing. After being attacked in the same way, by the same enemy, thirty times in a row players will get bored unless the designer makes sure something different happens. The sole motivation for continuing to play shouldn't only be the enemy difficulty or the fun of attacking them. It should also be 'what's around the corner?'. Don't let the player predict that.

Scripted events can be anything. They could be something simple like an enemy suddenly jumping through a window, or blasting through a door. It could be a surprise attack by a group of cooperating enemies, or a sudden change in the environment like a collapsing bridge or exploding building. It could also be as complex as an event that drastically changes the course of the story or gameplay like killing off an important character or revealing a suite of new objectives in a suddenly flooded area. Here are some examples of scripted events from well-known games:

The idea behind scripted events is to help players enjoy the game, add extra life to levels, and to provide variation. They pull the player into a believable world and keep them entertained and impressed by what they experience.

- In *Unreal*, the player is trapped in a corridor. The ceiling lights progressively go out one by one and then a door opens and the player is confronted by an important enemy (Skaarj) for the first time.
- In *Unreal 2*, a Skaarj jumps down on top of an elevator the player is in and rips open the ceiling.
- The Black Mesa Lab explosion at the beginning of *Half-Life*.
- Several areas in *Half-Life* where soldiers are brought in by a dropship.
- In *Half-Life 2*, during the hovercraft section, a big factory chimney falls across the player's path.
- Also in *Half-Life 2*, Father Gregori in Ravenholm helps the player at various times by shooting enemies.
- Sometimes in *Call of Duty 3* enemies block a door when they see the player approaching.

Another aspect of scripted events to keep in mind is that they help the game world feel like a larger place. They help the player forget that the game world revolves around them. Events don't always have to involve the player. The player can simply be a spectator. Scripted events that are completely irrelevant to the story can add just that extra bit of depth. A plane crashing into a mountain in the distance may not have any significance in terms of the story, but it is impressive to see and helps the world feel more expansive.

Events must occur in the player's view or else they haven't happened. Telling a player a spaceship has crashed is no fun. It is fun if the player experiences it themselves. The player should feel like they are part of it all. In *Unreal* most of the game's big events had already occurred by the time the player got to the various areas. While this fit that particular story well, it also left a dearth of impressive events for the player to experience while they traveled through the world. In contrast, in *Half-Life* events usually happen while the player is there, and it makes a big difference. More immersion and spectacle becomes possible. A player shouldn't have to read about something that happens. It should occur in front of the player's eyes.

Yet another reason to implement scripted events is to introduce new traps, enemies, items, or gameplay changes. If there is an important new type of enemy or a very dangerous trap, a scripted event can be a powerful design tool that can communicate information to the player. Previewing the danger or uniqueness of an item or area can hand information to the player that they'll need once they enter that space. *Gears of War* did this very well in the PC edition when the large Brumak monster is shown to the player several times over several levels before the player finally gets to fight it. Scripted events can build up 'hype' for an upcoming showdown or other significant event. The Brumak would not have had the same emotional apprehension had the player simply rounded a corner and come face to face with it.

Weapons can be foreshadowed by forcing the player to face enemies already equipped with a new weapon so its power and potential is understood by the time the player gets it. Or, as in *Unreal's* Eightball cannon, hyping up the weapon through the story and making it come across as a mythical weapon hidden somewhere in a hard to get to place. Finally, it can be achieved through less complicated means as in *Half-Life 2* where the player was directly given the Gravity Gun through a series of cut scenes/conversations. In any case, incorporating scripted events into gameplay-significant introductions have a much more positive effect than just hiding a new weapon somewhere in a level.

This same idea can also be seen in other media in addition to games. For example, in the movie *Aliens 2* the marines are not confronted by the aliens immediately. Instead they

are introduced to them step by step; first by the girl, then the motion sensor, then through Face Huggers, and then finally through a big exposition. Many games, movies, television shows, and other media show new enemies by the 'after-effects' of their presence before making a 'proper' introduction. Scripted events are the tool in the designer's pocket that can incorporate this type of foreshadowing into a game.

FLOORPLANS

Floorplans in singleplayer are simpler, to a certain extent, than in multiplayer. Contrary to multiplayer, long corridors and dead-end rooms in singleplayer can be acceptable and are sometimes even interesting since they can be used to enhance theme, style, and/or atmosphere. Unlike multiplayer, singleplayer relies more heavily on enemies, story, scripted events, and atmosphere to accentuate the floorplan. What scripted events happen and where? How well can the AI behavior adapt with different floorplan varieties? What is the atmosphere like? What is the story at this point and how does the floorplan reflect it?

Often, specific needs must be met to match specific events, for example a very large room or a dead-end corridor. As a result, the actual layout is already roughly conceptualized which makes the designer's job a little easier since they can do what they like within the gameplay and story constraints.

Even with this freedom, a few typical layout elements could improve the game experience and/or lessen the designer's workload, like reusing areas. Don't be afraid of reusing existing areas from the level. The work will get done faster, and it adds more depth to the level by reinforcing the theme throughout different locations in the level. Just don't overdo it and be consistent. There are various ways to reuse areas.

• The Preview Method

The preview method allows the player to see areas that currently are not accessible but will be later in the level or game. This allows a designer to reuse some parts of the level and also provide information to the player. This means less work for the designer and it also benefits the game experience. It creates the illusion of a world larger than the immediate play area and can also help the players orient themselves. The player can identify landmarks and understand how the games areas interconnect.

In the screenshot on the right, the player travels over a bridge crossing an underground river and can see a little beach in the distance.

About 15 minutes of gameplay later, the player actually ends up on the beach that was viewable from the bridge.



Image GP 14



Image GP 15

This can also be done in an even simpler way. *Half-Life 2* previewed a large skyscraper in the distance at the start of the game.

By the end of the game the player actually reached and entered the building.

The final location of the game was previewed from the very beginning of the game.

• The Evolution Method

Another way to reuse level elements is the evolution method. The player returns at least once, sometimes more, to the same area in a level. But each time they return, they are presented with an altered environment or change in the gameplay. One example is from the *They Hunger* mod from *Half-Life*. In one of the levels, the player infiltrates a large building. Later, the building catches fire while the player is inside the same area, but now with flames and scorched walls and objects surrounding them. A few levels later, the player is again returned to the same area but at this point the fire has gone out leaving an altered environment that changes the floorplan due to the blackened husks of objects and new holes in walls and floors. The environment is played several times, but each time there's a significant difference and thus does not become boring. Changing the visuals and gameplay each time helps immerse the player in the game world as the changes demonstrate that the world is alive and subject to change – just like in *Real Life™*.

• The Recycle Method

The last method simply reuses and recycles the area with little to no change. While this may sound like a cheap way of extending gameplay time, there are ways to use it well. One of the simplest methods is to send players to the left (or right) side first in order to perform an action, for example to hit a button that opens a door on the right (or left) side. This causes the player to cross the central area multiple times.

An example of this in a game can be found in *The Darkness*. It used two subway stations as hubs to get to nearly anywhere in the game world, effectively reusing the exact same level a dozen times without degrading the quality of the game. Instead it enhanced the game.

Half-Life expansion packs *Opposing Forces* and *Blueshift* also implemented this method on a much larger scale. In these packs, the player plays either a soldier or a security guard in the same area and during the same time frame as the story and events from the original *Half-Life*. Often the player would walk into the same rooms they saw when playing as Gordon Freeman in the original game, but are instead placed in completely different situations and they view the game story from different perspectives.

Reusing areas can greatly reduce the amount of work on an environment. Effective reuse can be a positive in terms of both the depth of the experience, as well as for the length of the level, and even in terms of the amount of work for the artist or designer. Just be sure to always have something new whenever this method is used. Reduce, and remove any predictability the player may experience by returning to a location they've been before.

One can also build multiple unique variations for a single area. While it may mean more work overall, it is nice to have multiple routes to a goal each of which have their own advantages and disadvantages. This is something that is always possible to implement, even when working on a very linear level or game. Two different exits from a room which eventually lead to the same area is the simplest example of this method. Another example would be a more open floorplan where most rooms have several exits and all connect to the same set of corridors. Games like *Hitman* make very frequent use of this type.

By now, it's probably fairly clear that I'm not exactly a fan of dividing gameplay and visuals between different people and departments. As described previously, I have the same opinion regarding singleplayer floorplans. When a floorplan is designed, it shouldn't only be fun, but also artistically and technically feasible and cohesive. I have had singleplayer floorplans come across my desk that were huge and thus impossible to create in the given time restrictions. I've seen floorplans with giant open areas that were thus impossible to optimize. I've seen floorplans that required skyscrapers in a jungle which rarely makes artistic sense. I've even seen some that threw out all the rules of composition and requested that all the important elements be placed in one small area. I could continue but the point should be clear: basic insight of each element is needed even if one is responsible for creating only one particular aspect of the whole. When making a floorplan, endeavor to make it realistic in terms of workload, technical limitations, and composition – not just in terms of gameplay.

LANDMARKS

The primary purpose of a landmark is to help players navigate through the world. They can be large and small, they can be buildings or rock formations, or they can be small and unique like graffiti on a wall, or even a unique vehicle parked somewhere. The key is that they stand out in some way. They help people remember locations in the overall map and grounds their spatial awareness. Landmarks are important to gameplay because they give the player direction. Getting lost in a level is always frustrating and landmarks can lessen this possibility.

Landmarks also can help players get a feel for the scale of the world and preview their destinations. As mentioned before, the giant skyscraper in *Half-Life 2* is a perfect example of both. Players saw their journey's end right from the beginning and the tower in the distance also gave them a sense of how large the city was. This, in turn, helps the environment appear more credible and realistic. *Oblivion* used its central city's tower in a similar manner; it gave the player a consistent landmark to look for so the player could orient themselves almost wherever they went in the level. Many games make use of smaller landmarks, for example *Bioshock* used many small unique elements to help give each room an individual feeling. They broke each of the areas into sub-areas for the player which was enough for basic navigational needs. We'll return to this more when we examine how composition can guide the player in a certain direction.

INTERACTIVITY

Interactivity makes the world more believable and it can even add fun. Interactivity means that something influences something else. This could be something reacting to the player's actions, or simply something as simple as vegetation reacting to wind or water. A world is not a static object – it lives. Videogames differ from movies and other traditional art forms through their interactivity and animation. Yet those things that live and breathe and move in life will not do so in games unless designers set them up to do so. Allow the player to talk and interact with NPCs, allow them to make changes to the game world and allow NPCs to react to such changes. Make the vegetation sway in the wind, have the characters leave footprints behind, show that the world is changeable or at least demonstrate some sort of activity. Practically anything that can be done to remove a static feel is a change for the better. 'Every action has an equal and opposite reaction' is still an incompletely attained goal in nearly all games and designers have a direct influence on this so endeavor to increase the amount of movement and life.

Scavenger Gameplay is one of the most widely used interactive elements in singleplayer gameplay. Scavenger gameplay is the name I use for the common 'key-quest' where players must enter area A in order to push a button or find a key that will open or unlock area B. This method can quickly become boring or a cheap gameplay ploy so use it sparingly. Avoid being overly simplistic when implementing this type of gameplay. Running into a room, grabbing a key, and running out simply does not challenge a player. Try to make use of secondary elements in conjunction with the scavenger gameplay, for example the player must save an NPC who will then reward the player by opening the door.

A more complex example could be one where the NPC who holds the key is actually part of a mission/quest. Perhaps the NPC falls off a cliff and the player must then retrieve the key. The player would need to find a safe path down, probably fight off some enemies along the way, maybe in a sewer system, and then the player would need to retrieve the key from the NPC's body. This would also allow the designer to reuse the path down as the player travels back up which would be an efficient use of the time invested in creating the area. Try to create unexpected situations for the player to pass through in a variety of ways. Whenever the player feels like they can easily predict what will come next, try to surprise them and make something different happen.

Keys and buttons can take the form of many different shapes and sometimes functions as well. In addition to regular keys, the player could search for a battery, a key-card, an access code, a power switch, a control panel, a person, a tool, explosives, or even manually trigger an environmental event to open the path, like boulders crushing a door for example.

The purpose of scavenger gameplay is simply to encourage the player to explore and achieve something else first. The more creatively this is done, the more the player will appreciate it. Just make sure there's enough variation in this throughout each level and the game. On larger scale, scavenger gameplay could translate into a situation where the player must gain a certain reputation with certain factions or teams by completing enough missions before they can access certain tools, information, or even a new gameplay space. This, in turn, can expose the next stage of the story. Games like *Grand Theft Auto* and most RPG's make use of this.

TRAPS

Traps are commonly used in singleplayer games and can be set up either by using environmental objects, enemies, or both. The most difficult part of creating traps is balancing them appropriately. They must be equal parts challenge, fair, risky, and rewarding. Often designers will place 'instant-death' traps with no warning given to the player, which is very frustrating. Remember: players want to feel as if they have some amount of control over what happens. Forcing them to die without allowing them to understand why, or even see the danger ahead of time and be able to plan around it removes control from the player which in turn causes frustration and irritation.

Communicate the danger to them – warn them. Place dead bodies around the trap area, or show an NPC or teammate getting killed by the trap, or place a suspicious detail in the environment nearby which could be as simple as a sound. Just don't allow the player to learn about it by using their saved game slots. *Oblivion* implemented trap communication quite nicely by clearly showing trigger ropes near the ground in front of the trap area. *Oblivion's* traps were fair because the player could locate the trap without running into it first by noticing the rope, yet the traps were still somewhat hidden and deadly enough to be challenging. A player should understand that when they died from a trap, it was their own fault and that they were in control enough that they could have avoided it if they had not made a mistake. Skill and leveraging the core gameplay makes games challenging, not luck.

ITEMS

Players should ideally always have just enough health/mana/ammo/etc. available. If not, the game becomes frustrating and besides, what's the point of having cool weapons if the player can't use them if they're always out of ammunition? On the other hand, provide too much of an item, and some of the tension and challenge of a game is lost. As with everything else, endeavor to strike a balance. A corollary to this is variation. Encourage the player to make use of the different items, for example weapons, by scattering them, and their ammo, throughout the level. This depends on the game, of course, but if the player carries five weapons with more than enough ammunition for all of them, then they will only use the best weapon and they'll ignore the other four. When this happens, there's almost no point in having created the other four. A designer could even set up specific areas that leverage the strengths of a particular weapon. For example, creating a section where a sniper-type weapon is particularly effective, or an area where a close-combat weapon would be more effective than others.

One common mistake many designers make is to spawn new monsters/enemies immediately after the player grabs some health or armor. Unless specifically called for by the game design, try to avoid doing this. Few players will attempt to grab that 25 point health pickup if by doing so, they expect to lose those 25 health points again when a monster spawns behind them and attacks. This is a pointless gameplay feature that can easily lead to frustration. Giving the player a reward and then punishing them for taking it is a cruel practice.

There are certain exceptions for special types of health and armor, however. If the health pickup is a larger quantity than usual, and the player will walk away with more health than they entered with, then giving them a challenge when they pick it up is generally allowed and, sometimes, encouraged. The difference between the two situations is that in one, the designer is giving the player a certain amount and then immediately taking it back. With the other, the designer rewards the player with something large, and the player can keep some of it, but it's up to them as to exactly how much extra they leave with.

Another use of items is that they can be used to indicate paths through the environment. An item can be placed in such a way that they players receives a hint as to a direction of travel. If the player must choose between a left and a right corridor, and the right corridor has a few health pickups, the player is more likely to be drawn to the right corridor than the left one. Another example would be a situation where the player is up on a ledge with another ledge below them. If an item the player can pickup is within view on the lower ledge, then the player gets the message that it's an ok place to jump down to. If there were no item there, the player could become stuck because they would not know that it's safe to take the jump down.

DIFFICULTY

A level designer's goal is not to frustrate players, but to entertain them by offering them a fun, challenging, memorable experience. Challenge and frustration are two different things and frustration is generally not considered fun by most people.

The modern gamer wants to experience action and adventure. They don't want to be forced to overcome overly difficult challenges that demand weeks of time to solve. As gamers grow older, an industry trend, and start building families, they have less time available and want to progress at a pace that allows them to experience a fair amount in a shorter period of time than what they used to need. If the average Joe comes home after a hard day of work and at the end of the evening has just an hour of free time, then the last thing they want is to get stuck just ten minutes into playing. In such situations they get frustrated instead of having fun and they turn off the game. Remember – games are about entertainment.

As a level designer, the levels created should support different difficulty settings the game offers. Games like *Devil May Cry 3* are ridiculously difficult even on the easiest skill level settings. Different difficult modes allow players to decide how intense and challenging they want the game to be. Don't force your own opinion on difficulty on them.

There are many ways levels can be made easier on low difficulty settings

- Weaker enemies - less health, do less damage to the player
- Fewer enemies
- Better weapons found more often
- More (good) ammo
- More ways to recover health in more places
- More cover

It is important to avoid making levels more difficult by removing save checkpoints. The difficulty of a level should be contained within the gameplay – not in the number of save checkpoints. Removing them will only lead to frustration where the challenge should rest in the skill of the player versus the enemies.

Also, checkpoints can be used to communicate that a significant event will probably happen soon. The player will be more likely to suspect that something big will happen near checkpoints. If the gameplay in a level is built on suspense, checkpoints can easily be figured into the design to enhance or allay tension. Or they can be used in a way that goes against player expectations just to keep the tension that much tighter.

PACING

Pacing is used to describe the ebb and flow of experiences that the player has as they traverse the level. Distribute interesting things throughout the entire level. Make sure to provide enough to keep the player interested, but don't overdo it because there will be nothing left to use at the end of the level which should be the climax – the part where the coolest/most challenging/most awe-inspiring thing happens. If possible, attempt to preview the interesting parts of a level right from the start. For example, when a player enters the level, a cutscene previews events that take place near the end of the level, or maybe some key areas of the level. Wrap it all in a nice package and present it to the player in pieces. Another example could be that, upon entering the level, the player sees a very impressive building in the distance. Instead of traveling right to it, they could pass through lesser areas that are still interesting, like maybe the foundations of the same kind of building, until the end of the level, or suite of levels, where they finally reach that grand building they saw all the way at the beginning. Introduce the cool aspects of a level straight away in order to pique their interest and get them hooked on playing, with the knowledge that more is to come.

Games often do this on massive scales. Plenty of games, like *God of War*, show the player what they will attain, and then they take those features away from the player. As the player progresses, they earn those features back. Previewing cool aspects can help encourage players to continue with the knowledge that they'll get something interesting in return.

Boss fight levels are especially notorious for this approach. The PC edition of *Gears of War* did this very well by revealing the huge Brumak several times through several levels. It kept the player playing because they knew that they would have to face it at some point, they just needed to play a little longer to get there... and a little longer... and a little longer. Motivate the player to continue their progress, just be sure to balance the length of the effort with the reward.

FINAL THOUGHTS

The best result will, in the end, be a combination of all of the above in the right balance. The more of these that are allowed to mix together and influence each other, the more coherent the world will end up being. The whole is greater than the sum of its parts. For example: a player runs through a corridor when suddenly an enemy opens up some water and steam pipes. Water floods the corridor, and the player ends up with reduced movement in waist-high water with their vision obscured by steam. After some time, perhaps the floor collapses under the weight, or maybe just a portion and the player is sucked into a hole. The player then ends up in a completely different area and will thus need to find their way out.

The different aspects described in the above situation are: enemy behavior variation (enemy performs a special, unexpected move), environmental depth and interactivity (world can be damaged and damage has a result), environmental visual changes (the flooded corridor looks different), and last but not least, there are significant changes to the gameplay (gameplay speed, steam obscuring vision). Also, once the floor collapses, the player gains a new task and enters a new area, which is always somewhat of a reward as it encourages more play.

These types of advanced events happen rarely in today's games. Break the mold of the standard 'enemy shoots at player' gameplay and add some much needed life to the level and the gameplay!

One of the most important points in this chapter is to inject variation. Do the unexpected! The player should never be able to completely predict what will happen or otherwise the game will become boring. Repeated gameplay tools, like spawning monsters in a certain way, will only have an impact the first few times the player sees it. After the 20th time, it becomes part of the background. Adding variation to all these systems will create a more cohesive, living environment, keep the player on their toes, and keep them playing longer.



AUDIOVISUALS



AUDIOVISUALS

INTRODUCTION

Just as gameplay is important, so are the visuals and audio. The appearance and presentation of a level render it attractive to play. Looks grab peoples attention and will start them playing but the gameplay will keep them playing. Thus, a balanced mix of both is necessary for a successful level.

Visuals in levels are very complex because they consist of many different, and interdependent, systems. A level designer must retain an overview of all the different pieces if they are to succeed, and it is this that is the most difficult to master. Each change and addition or subtraction will impact the rest of the level and potentially throw it out of balance.

If done correctly, a level is a form of art. The audiovisuals have a higher purpose than simply communicating the theme to the player. An important part of art is that it conveys feelings and emotions. Art isn't defined by mathematic calculations; it's defined through the artist's emotion. A level should communicate a certain emotion, or range of emotions. It should scare them, or make them happy, make them feel cold, or warm to match the environment, etc. Creating a level, at its base level, is about communicating a time, a place, and the emotions that go with them to the player. Make the player experience all of these!

Designers often make the mistake of finishing a level too quickly. A couple of crates over here, some big lamps here and here, some buildings around the outside, and voilà! The geometry may show the player where the walls and floors are, but it fails to communicate emotion or create any sort of atmosphere. It ends up feeling like a collection of shapes in a generic space. This will make the player feel like they are just sitting in front of their monitor or TV set playing a game. Instead, they should really feel like they are the space marine in a military outpost being attacked by aliens. Simply put, it ruins immersion. The eighties and nineties have come and gone along with moving dots and the simplest of possible geometry. Games have become more complex, are capable of so much, and are steadily approaching photorealism. The player should feel the coldness of the place, the anger in the character, the pain of the loss, the darkness of the labyrinth, the warmth of the sun on the skin, the sea spray in their face, and the smell of the sewers. The graphic potential is there – use it!

This chapter will examine how a designer can best make use of visuals and sounds in an environment; from architecture to lighting to sound. What to do, and what not to do – for example, why certain colors work better in situation X and Y but not Z.

COMPOSITION

INTRODUCTION

Composition is a broad subject. It is important in every creative work and people have written entire books about composition as regard photography, painting, and other artistic mediums. Mastering composition can take years but, all the rules aside, at the end having an eye for it is still important. The designer should be able to sense when it is good. This sense can be enhanced by teaching, but some seed must still be there first.

Composition generally refers to the balance of position and intensity of the visible elements that then guide the eye in a direction of importance. Does the architecture feel balanced or does it draw all of the focus to one corner of the room? Do the light colors complement each other well or does one of them grab too much attention and throw the balance off? Composition is an abstract notion. It's complex and difficult to define. In level design, it is particularly difficult because the composition should work from as many camera positions as possible, but in particular from the most important angles, for example from the entrance to the area.

Many of the individual aspects of composition relative to level design will be explored in the various visual chapters. This chapter will limit its examination of composition to the general rules.

THE BASICS

Practically nothing is placed completely randomly – not even natural objects like trees, bushes, and rocks. There is always a certain reasoning behind the positions and angles of the objects, even in Real Life™. How does a designer arrange natural objects to look truly natural? How does a designer prevent a room from concentrating the focus too much to one side?

When unsure if the elements are unbalanced there is a technique from the traditional arts that can help. Squint while looking at the work. If something clearly stands out more than other elements, and it isn't intentional, like a landmark, then it's probably unbalanced. Another technique is to step back and look at it from a distance. Actually get up, and back up anywhere from 5 to 15 feet, depending on the size of the screen. If something stands out that shouldn't, then there's an imbalance.

Examining how other forms of art and design handle composition can be of great assistance. Take photography, for example. The difference between level design and environmental photography is that the photographer must go out and find a pre-existing composition – they only have to recognize it and then get the right angle. Level design does this in the opposite direction. The designer must create the composition, and then ensure that all the photographers can come in and take photos of good composition from any angle. Designers must pre-visualize the area and then create it in order to create a good composition. If the artist knows what to look for, then they will also know what to create. For example, S shapes often work well in photos and they work well in levels as well. If a designer knows that S shapes will result in good composition, they can then keep that in mind while constructing the level's base and implement that shape into the environment.

As will be described in more detail later, flow and balance are important. This can be illustrated through the following examples using scanlines. If all the elements in a scene lack contrast, the graph of the scene looks flat. The lack of contrast creates boring environments, and the lack of contrast in gameplay creates boring gameplay. It is a dull, flat line where nothing significant happens. All the rooms are the same shape with the same textures and the same light colors and brightness.

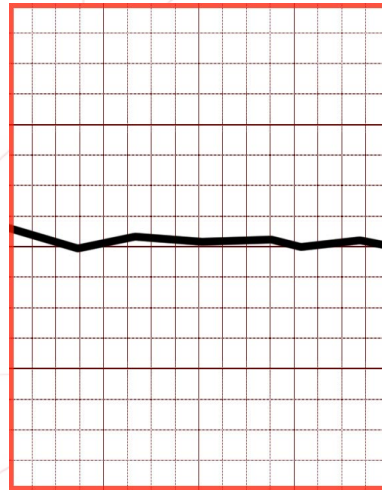


Image AV 1



Image AV 2

If there is a difference, a contrast, in the area, for example an orange light in a very blue room, the line would suddenly break the flatness:

However, if the same contrast is always used – the same orange light dotted around everywhere – it would become too repetitive. Variation is good, but if you repeat that variation too much, the variation itself will become repetitious.

On the other hand, if the difference, the contrast, is too large, the graph will have very steep lines. It won't flow anymore and the eyes will hit a wall. This causes imbalance.

This situation could be caused by many different situations:

- One side of each room has much more detail than the other side, which is an overly simplistic arrangement.
- One side of each room uses many different colors and the opposite side does not.
- One side has a very bright or dark spot.
- One side is very open and the other side is very closed.

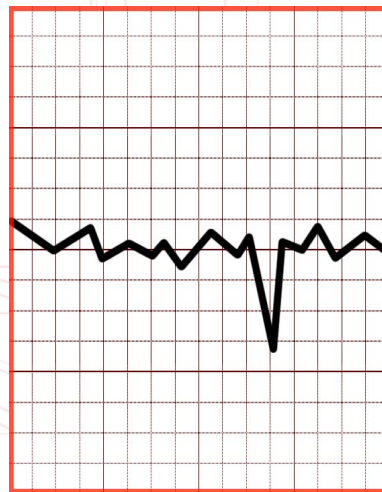


Image AV 3

Having a visual or artistic discrepancy between two sides of a room does not necessarily represent bad composition, especially if the intention is, for example, to help the player navigate the area. It is only a problem when it occurs unintentionally and contributes nothing to the level other than a lack of balance.

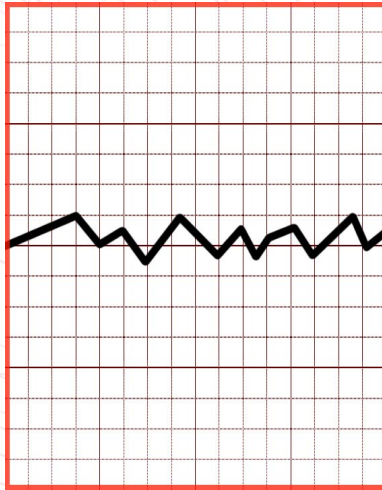


Image AV 4

Good composition contains contrasts that are strong enough to represent a change, but not so dramatic as to cause the eyes to hit a wall. A gently flowing line that is neither repetitive, nor flat, nor too different: a balance of all of the above

The goal is to create an area that offers enough contrast to change the area, but not so much that it feels completely different. The right balance is key. The 'more is always better' rule does not apply to composition. Contrast in composition means using variation in intelligent ways. Is the contrast in the level just a variation or is it a theme in and of itself? If it is, then what is the purpose of having it in the scene? Does the area need more than one style and/or theme? If they do, then can they blend together smoothly enough to keep the balance and prevent a too-strong contrast?

STAY ON THE PATH

Composition should also aid gameplay. The composition in an area should shape the area in order to show the player the path forward. If a corridor has an exit on the left, the composition should not lead the player to the right.

For example, when the exit on the left side is hidden in darkness, yet the unimportant right side grabs the player's attention with its bright lighting, the player is likely to miss that exit. Unless the player knows the level, they will be drawn to the right each time until they recognize the area and remember the exit is to the left. Important objects and paths in the level should attract the player. Use composition to guide their eyes toward these places. Don't allow the players to hunt for them. Unless required by design, the risk that a player will miss an important object should be avoided. Players in multiplayer games get very frustrated if they run down the wrong corridor while being chased simply because the real exit was 'hidden' from them by bad composition. Even in singleplayer it can be quite frustrating for players to walk around for hours looking for the one little item they need for a quest.

As with paths, important NPCs and items should not be placed in obscure dark corners. The room or area serves the NPC's or object's purpose, not the other way around, usually. A large landmark such as a building or mountain should not be placed behind other, larger elements. The other pieces would reduce the attention and power from the important object. They are supposed to support it by guiding the eyes in the direction of the object.

COMPOSITION OF NATURAL OBJECTS

Natural objects such as trees, rocks, and especially vegetation should be placed in such a way that a balance is struck between being placed at random, and placed too artificially. True randomness often results in a messy and even unnatural look.



Image AV 5

In the good example, the different elements support and rely on each other. The plant and grass needs the rock. In the bad example they are placed randomly and do not appear to have any relation to any of the other elements. The good example demonstrates a certain system and order in the way they were placed, which is realistic. After all, the rocks were probably there before the plants and grass. In the bad example one cannot determine if there was any order in the landscape's development.

In the bad example, the two largest rocks are also placed right next to each other, where as in the good example they are spread out more over the area, balancing it out. This also goes for the plant and the grass, which is nicely spread over the entire area in the good example, but only on one side in the bad example.

MOVING GEOMETRY

Moving objects are one of the best ways to grab the player's attention. Pieces of the level that move, like a machine, is bound to catch the player's eye even if it's off in a dark corner. Therefore, a designer can use moving objects to balance composition by giving an empty side of a room a moving element to balance the more complex opposite side.



Image AV 6

Additionally, a designer could use moving objects to accentuate more important areas, for example a machine near an exit door or a windmill in the distance that marks the player's destination. Try to think outside the typical mindset when creating moving level objects. Machinery is an obvious and oft-used type. However, moving objects can also include animated lights, like warning lights, broken flickering lights, or particles like steam, or items blowing in the wind like flags and banners. Even an NPC can serve as a piece of animated geometry to push the composition in a certain direction.

ATMOSPHERE THROUGH COMPOSITION

The way objects are placed and shaped will partly determine the feeling and atmosphere of the level. For example, if a fight takes place in a house then afterwards the rooms should look like a mess and possibly be damaged as well. Thus the objects should be placed somewhat randomly. Placement of the objects supports the theme. Without believable placement and composition, there is no support for the theme.

A more complex example would be if most of the light sources are located near the floor in order to give the room a scary or depressing atmosphere. The low lights would make the player feel like they are being pulled toward the ground. If the lights, by contrast, were near the ceiling, then the player would feel more uplifted which would be the incorrect impression for the atmosphere and composition.

These same concepts are applicable to horizontal and vertical architectural alignment. This alignment refers to the direction of the composition. If a room contains many horizontal lines, then there is a horizontal alignment and composition.

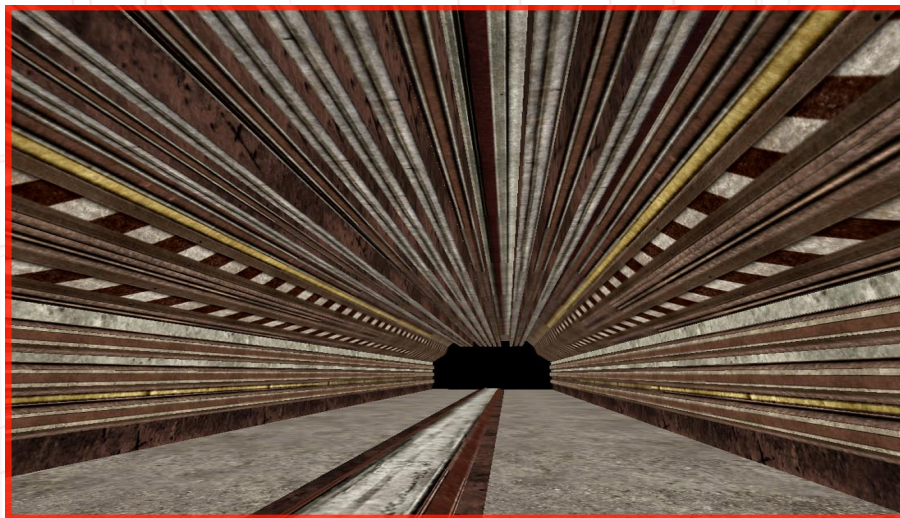


Image AV 7

On the other hand, many vertical line will contribute to a vertical alignment and composition.

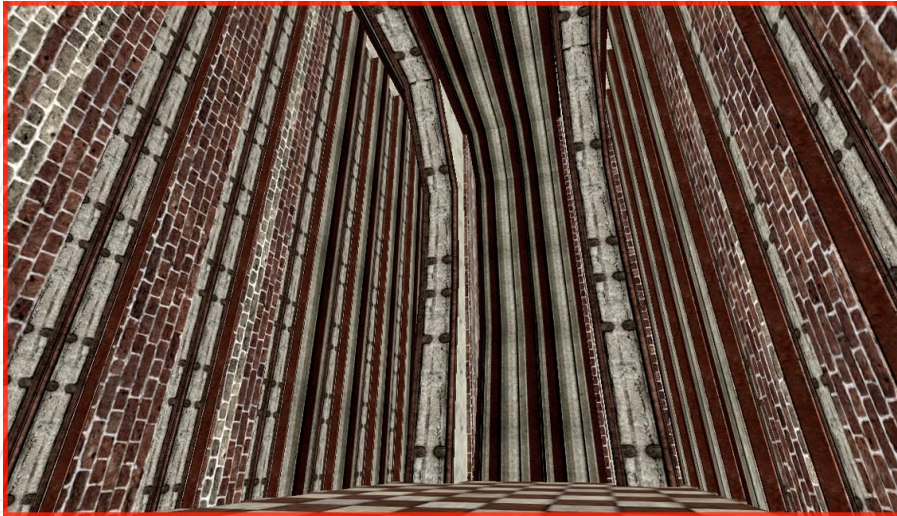


Image AV 8

The thinner the lines are, and the more there are, the quicker they will create the illusion of the horizontal or vertical alignment. Now, why is it important to understand and control this? It's important because the alignment supports the theme and style, and therefore the overall composition. If the goal is to create a large and impressive wall, then it would require primarily vertical architectural elements – not horizontal.

In the following example both walls are exactly the same size. In fact, their both made from the same base brushes.

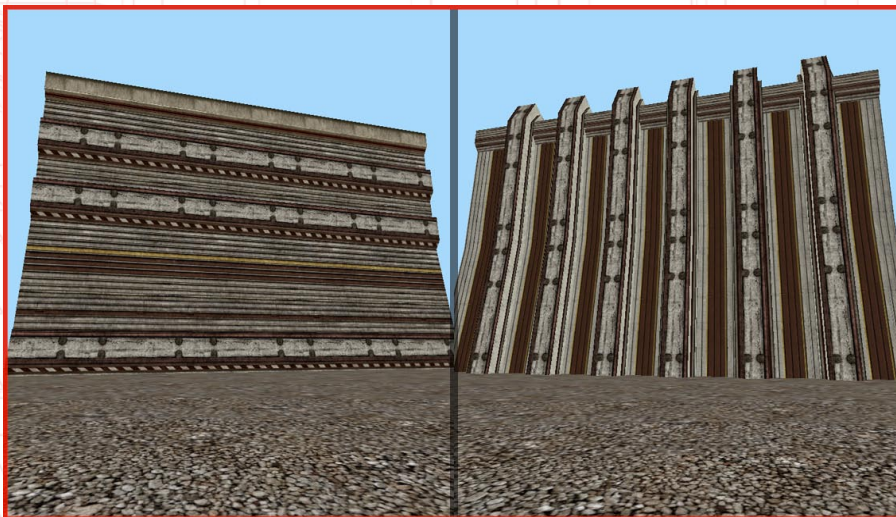


Image AV 9

However, when looking at them, the right wall looks higher than the left wall.

A tower is another typical example. To emphasize a tower's height, a designer can add vertical elements to the tower sides in order to give the player the impression that it's taller than it actually is. The proper alignment can make a significant difference to the perceived perspective. *Oblivion* made good use of this with the tower in the center of the capital city.



Image AV 10

Vertical elements can also enhance a feeling of coldness while horizontal elements can imply warmth. Vertical elements are generally considered more static, official, and impersonal, while horizontal elements are connected to personal, intimate, and more down-to-earth sentiments. When building a living room in a cabin, horizontal would probably be the wiser choice, whereas a living room in an old estate manor would probably use more vertical elements.

The orientation isn't only restricted to line forms. Other shapes can help express the orientation, for example arrows. In, again, *Oblivion*, there is a church constructed from many 'pointy' shapes which result in a more vertical and impressive appearance.



Image AV 11

Not only is the church surrounded by buildings with pointy roofs, and has pointy spikes on top, but there are also pointy arches above the windows and doors, and the church itself has almost stair-like structure which implies a higher perspective than it actually has.

LIGHTING

Lighting composition is very similar to architectural composition. Lights should be well balanced and well placed throughout the room so that there are no large dark gaps that can unbalance the composition. Avoid shifting the focus to one or two areas unless the story or situation requires it.

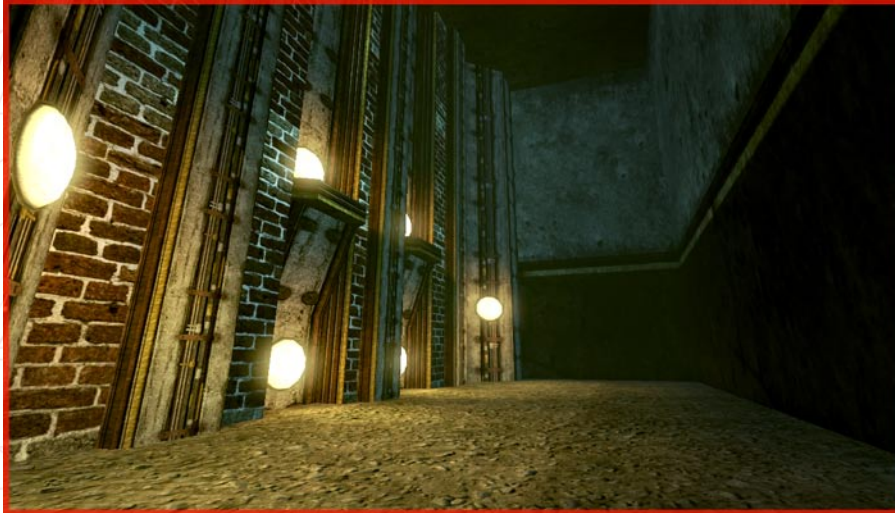


Image AV 12

In this example above all lights are located on the left side. Spread them out to help balance it – the same goes for light colors as well.

The light colors used should be varied and balanced. If all the lighting is one color, like blue, then the scanline is flat and the result monochromatic.

The colors should be balanced. There are, of course, exceptions to this, and some environments can require large amounts of blue lighting, but some variation within the blue should be present – as much as the theme and atmosphere can allow.



Image AV 13

Another aspect of color balance is that the difference between some colors can produce harsh contrasts. As explained later in the Lighting chapter, some colors are much stronger than others. Red, for instance, is the strongest and is so aggressive that it draws the eye's focus whenever present. It is for this reason that what, where, and how much color is used requires balance and composition. Adding many strong red lights in one corner of a room will unbalance the scheme unless the player's attention needs to be drawn to that area intentionally. One example could be that an area is being closed off and the player needs to notice the closing door. Add a red light that turns on when the door is closing would be a good implementation of this.

IN-DEPTH EXAMPLES

SAE

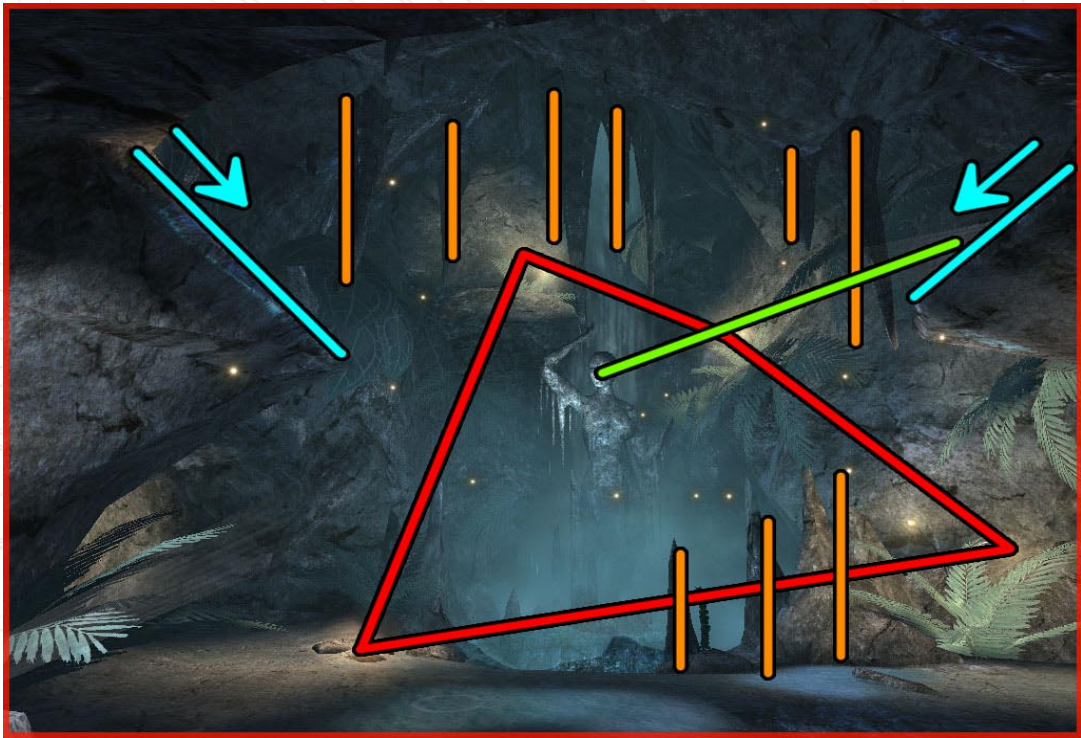


Image AV 14

Yellow lights are placed throughout the scene, ensuring that no area stands out more than another. If the four yellow light sources were connected, it would, more or less, cover the entire center of the image without being too artificial a shape (red line). The center receives extra attention from a brighter blue light that highlights the most important and central element of the area, the statue. In addition to the lighting, the stalactites add a vertical emphasis to the geometry, and make the whole more impressive (orange lines). The arm of the statue is vertically aligned as well, and points up. The two ledges in the foreground both point toward the statue, further emphasizing it (blue lines). There are two bright floodlamps on top of the two ledges, illuminating and highlighting the statue (green line).

KRODAN

The overall intention with Krodan was to create a very impressive scene. Thus, there was a strong emphasis on adding vertical alignment elements to the scene. While the general idea of a large cliff with a big house built onto it, with a bright sun above, is already impressive, subtle and careful placement of various elements exaggerated this feeling. The lighting composition creates an upward-pointing arrow with the four orange lights, and a blue light in the middle of those to balance it out (red triangle). The sunrays casting down from behind the rock point towards the player which, from this angle, makes the appear vertical (red lines). If the sunrays flowed across the screen from right to left, they would have been horizontal and therefore would have disrupted the composition. There is also a fair amount of ivy hanging from the rocks in the scene which contributes additional vertical elements.



Image AV 15

The verticality of the disparate elements was carried down even to smaller details like the fence with its vertical lines, and even the grass which is pointy and vertical. The area also contains lots of pine trees which are typically shaped like upward-pointing arrows. If deciduous trees had been used instead, the overall vertical effect would have been reduced. The house features eight drainpipes on its walls which again, supports the verticality of the scene. The house itself is somewhat narrow, but tall and is positioned higher than ground level. Additionally, there is a deep crevice behind the fence to further accentuate verticality. The large round window at the top draws the player's eye up even more. On the left side of the house is a narrow waterfall that drops all the way down into the crevice (blue line). On the right side, which is not visible in this screenshot, is a large window in the rocks which balances the waterfall on the other side (green line).

It's quite possible to go into depth with hidden symbols and composition. While the average player will not completely identify every single layer, it will influence their general stance on the product. Unconsciously, it will make a difference.

If these layers and the depth possible is of interest, consider researching the idea more. One example could be researching 'The Golden Ratio'. I personally find that this is too much work and forethought in order to do it justice. I usually try to eyeball the ratio but, in theory, it would be possible to lay out the entire level mathematically. It may sound like an insane thing to do, but the important point is that even unconsciously, it will enhance the level.

GEOMETRY AND ARCHITECTURE

INTRODUCTION

The geometry and architecture of a level determine a large proportion of the environment's style. They are usually the first portions of the level created when the visual work begins. Geometry is the base upon which all the other visuals are built – without geometry, there can be no lighting or texturing.

The geometry and architecture must be compelling to the eye, but also logical to the brain. The structure should appear solid and well-grounded; otherwise it will provoke an unbalanced feeling – unless that's the goal. Architecture is one of the visual elements that has the greatest amount of impact on gameplay. As mentioned before, without the knowledge of one, the other cannot be effectively created. As geometry and architecture are very far-reaching subjects that are very dependent on theme, this chapter will focus on the ideas behind architecture and how to implement the basics.

When the inspiration well runs dry, searching for reference photos, viewing galleries at cg art sites, or looking at textures and imagining what 3D shapes could go well with them can all help foster new ideas. Movies, games, and television can also be good sources.

Architecture is something learned best by copying and examining existing works for the first few months or years. It takes quite a while to develop a unique architectural style. Copy a style that is appealing from a level and when out of ideas, simply have another look and the level to see how it was handled.

COMPOSITION AND BASICS

One of the most common mistakes beginners make is to add very large, empty, flat spaces. This is something to avoid for the same reason as stated in the gameplay chapters. Large empty space -> repetitive -> predictable -> boring.

As seen in the example, the large flat wall is not very interesting to look at. Additionally, the texture tiling on the wall is very obvious.



Image AV 16

The same rule applies to using the same architectural element too many times in the same area as it is simply too repetitive. Knowing when to stop and add variation is an important skill.

The geometry must flow together. It should be designed in such a way that the composition is clean and leads the eyes across the area. As mentioned before, this can best be illustrated with a scanline.

If the area consists of many large empty spaces then the line would be nearly flat. There is no flow and the eyes are not lead around.

When only part of a room has a large empty area, then the line would be partially flat, as illustrated below.

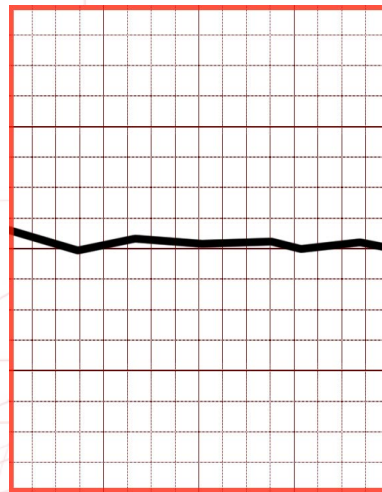


Image AV 17

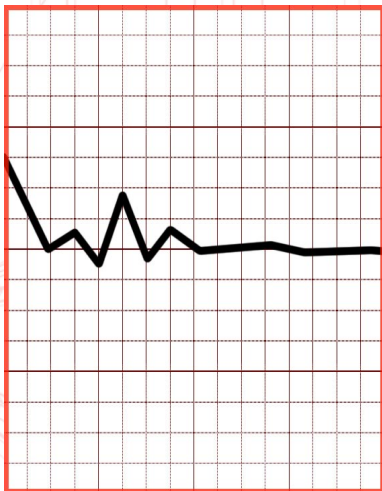


Image AV 18

The opposite can also happen. When there are too many large or special elements in the otherwise balanced room they will create strong peaks in the flow.

The player's eyes would hang on the steep walls of the peaks. The flow would be interrupted.

Adding such powerful or special elements is only permissible when absolutely necessary. In other words, when the player's eyes must be grabbed and their attention focused on a certain area. If this is not the intention, then the elements causing it should be removed.

The key is to balance everything yet ensure that there is still enough contrast: a line that is neither flat nor has overly strong peaks, as discussed before in the composition section.

In an ideal world no geometrical element would add too much repetition to the area; the amount of empty space is consistent throughout the area, and if there are large empty surfaces, they should never be so large that it becomes obvious that the texture tiles.

Let's return to the empty wall example. What it obviously needs is extra geometry to break it up; pillars for example. Flat surfaces are a major problem in many environments. While flat walls are often easily identified and touched up, flat ceilings or floors are often ignored. They should be broken up by differences in height and by other details.

Of course, it depends on the theme but surfaces should always be broken up by variation. Floors do need small variations in height. A good example would be to implement a few small stairs. Other options could be to add a few broken tiles, drains, steel bars, vents, holes, and so on.

This also goes for pillars and similar architectural details. Liven up their shapes - avoid simple cubes and cylinders.



Image AV 19

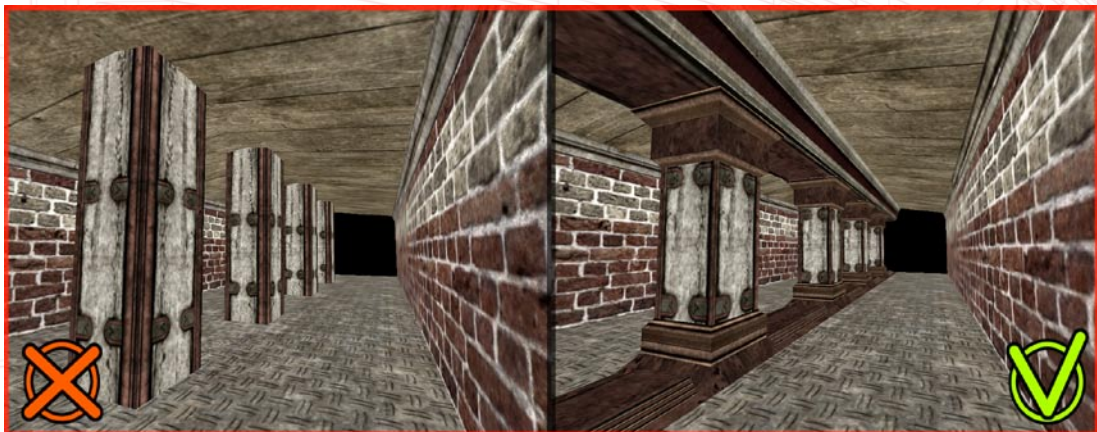


Image AV 20

Build them up based on multiple elements and/or add more support/detail to them. For example, show the player how the pillar attaches to the ceiling instead of just sticking it through the surface.

Another type of flat geometry to avoid is a large "2D" wall or textured wall. A flat wall with good texturing, but nonetheless still very flat and boring.

The wall still appears flat and boring because of the limited or non-existent depth it has. Adding multiple textures to a surface is a very good way to break it up but it is not a replacement for true 3D geometry.



Image AV 21

Trim is a border around an edge to add polish. This can either be done through good texture work, for example choosing a texture that contains a distinct border, or by adding extra geometry. Without trimming the sides of a platform, hole, shaft, door frame, or other similar features, edges will be cut off unnaturally. Adding trim eases the transition between two or more textures as well as adding extra detail to the geometry.

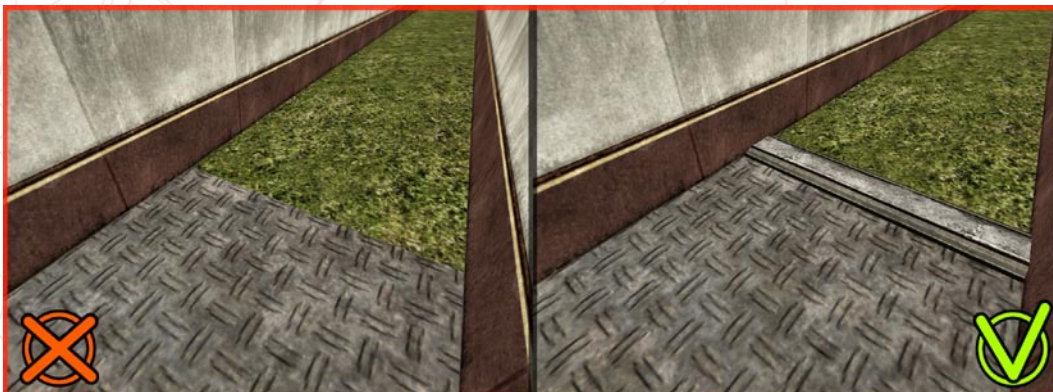


Image AV 22

Opening up shapes helps geometry appear more interesting by getting rid of the basic shape and it can make an object look like it has more depth.

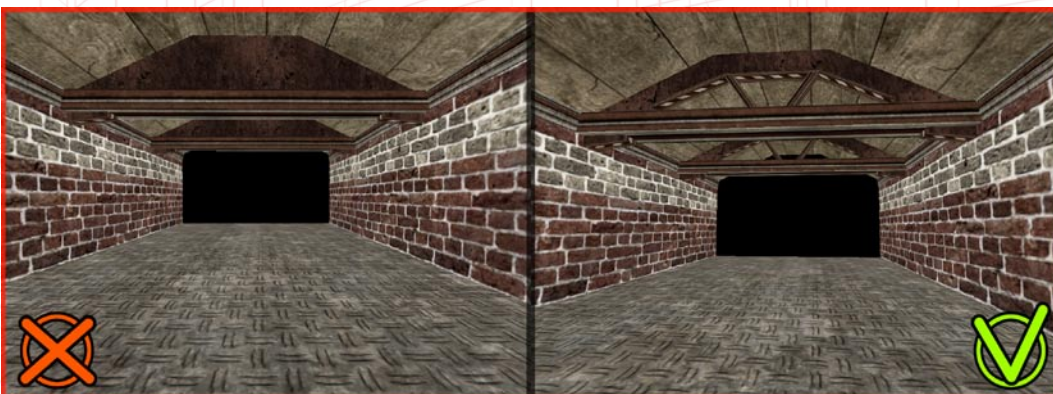


Image AV 23

UNITY AND CONNECTIVITY

Like everything else, architecture must convey a sense of unity and connectivity. A common problem is architecture that feels cluttered. There is a tendency to create a thousand-piece puzzle while the actual goal is to create a space that is simple, but contains the hint – the suggestion – of a thousand-piece puzzle. Stick to a style, theme, and the same level of detail at all times!

A potential bad example of style would be to create half of an environment using cubic shapes, but then finish it with spiky or rounded geometry. Two or more types of shapes in the same environment leads to a cluttered look and feel. A bad example of theme would be to mix two contrasting themes together with no compelling reason.

Finally, an example of disparate levels of detail would be to create a very detailed level, and then neglect the same level of detail in a few secondary corridors. Or the opposite – a few heavily detailed areas within a larger, simplistically decorated level.

All of those examples result in inconsistent levels. Remember that a single environment is being created; not fifty small ones glued together.

Connectivity is used to describe how the architecture interacts with itself. If one of the geometric elements is causing, influencing, or being influenced by another geometrical element then it should show that connection somehow. Connectivity is a very important aspect of the geometry. In order to achieve an end result with a united look it is necessary to show how all the different elements are built upon each other and how they're dependent on each other.

For example, if there is a canyon and a building is built against its walls, then that building will have specific design elements that demonstrate how that that building was made to fit into the canyon. It will not look like a regular house that's simply plugged into the rock wall. If, for example, the weather conditions are always rough in the canyon the house will have many structural reinforcements. The canyon and its situation determine the look of the house.

While this may appear quite obvious it is the key to connectivity and very often overlooked, especially on smaller scales. Different architectural elements often end up next to each other instead of being conjoined: pillars that don't feel as if they really hold up the structures they support, or pillars that seem to ignore the wooden bars running across the ceiling, and so on are a no-no. Different elements need to connect to each other in order to support the geometry. And most importantly, they should cement the feeling that one cannot exist without the other! Without the pillar the ceiling would collapse. The connection must be clear. Without the extra reinforcements the house would not be able to withstand the rough weather conditions of the canyon.

The maximum connectivity one can achieve is to make every architectural element flow into another element. A ceiling bar seamlessly connects to a pillar, which in turns seamlessly connects to a railing, in turn connecting to a door frame with great flair. When this is achieved, ultimately depending on the style, the result will be a great synthesis of architecture that not only looks cool, but also seems credible.

In this stylized example all the pillars, bars on the walls, and supports in the ceiling connect with each other and seem to influence each other by a slight bump whenever they cross another element.

Now this is a very specific style, but it illustrates the idea that one piece supports the other and all connect to each other. They demonstrate that they are not separate elements but that they belong to the same room.

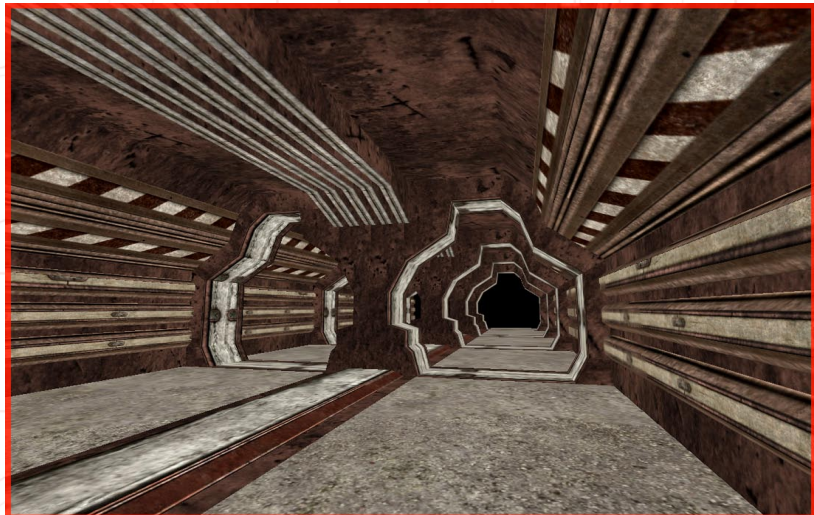


Image AV 24

The same concepts can be applied to more realistic styles, albeit in more subtle ways.

CUBISM

While the most used shape in environments is, without doubt, a cube (or similar) it's almost always a bad idea to throw it in the players face. Make sure that cubes aren't too obviously cubes.

Beginners often make two big mistakes. First of all, they create their geometry out of too many basic geometry shapes (such as the awesome cubes), and secondly, as a direct result, their environments look very blocky and lack finesse. Beginners often fail in sufficiently hiding the cubes they started with.

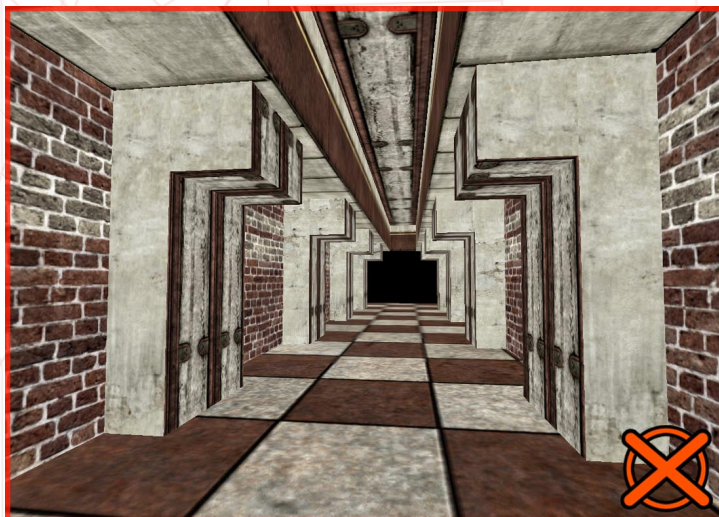


Image AV 25

In this example you can clearly see the basic shapes that started off the level. Hide the original shapes as much as possible. Some very simple actions such as chamfering a corner can do miracles. If the style is more realistic, try placing objects in corners and recesses, such as closets, which might help hide the basic cube

By making a few simple modifications much better results are possible. Break things up! As described above, chamfer corners;

round them off. Add a few extra elements, such as cables and pipes, to hide the corners. Ninety-degree corners have a tendency to look bad. Minimize the use of them unless required by the art style and the theme.

WEIGHT AND BALANCE

If the structure does not have enough support it will look like it's on the verge of collapse. A construction should always appear possible to the player. A structure that appears weak because, for example, a gigantic roof that is held up by barely any support is not only ugly but it also harms the game experience since the credibility of the environment is reduced. This goes for any type of platform or structure that seems to be floating.

The tower in the picture above could never be supported by the roof and the small building. Weak structures show the player that the world is synthetic which makes it harder for the player to become immersed in the story,



Image AV 26

atmosphere, and the rest of the game. This error is also often made in terms of the thickness of forms like walls. If a wall, or a roof, is too thin it will not look solid or strong enough.

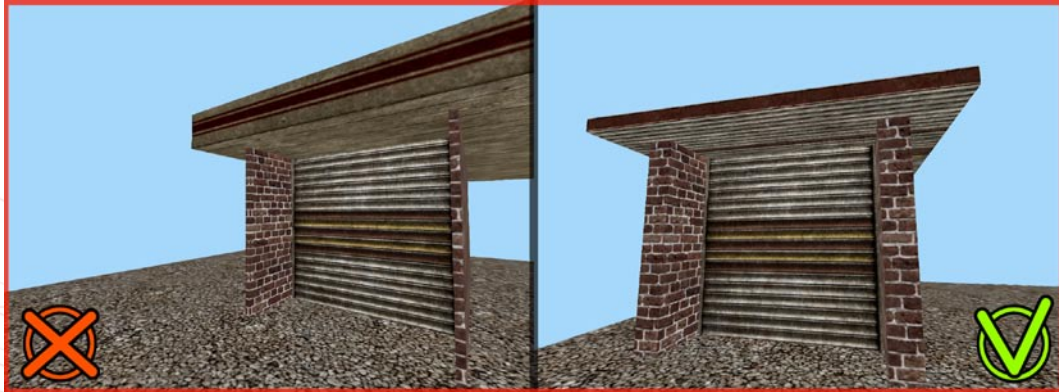


Image AV 27

In the bad example above, the building has thin walls and a very thick roof; neither are realistic. The wall thickness should be adequate enough to hold up the building and serve the building's purpose. A bunker or castle, for example, has very thick walls which should be communicated to the player.

Note that I said the structure should communicate solidity. I did not say it actually has to be mathematically and structurally correct. In game worlds everything is fake in the end. Perhaps the building doesn't even have an interior – it doesn't have to. What matters is that the player thinks it does if they look at it. The illusion of solidity and stability is what counts and it is the artist's job to create that illusion.

The appearance is always more important than the logic. Most people will not look for a logical explanation or argument if it looks like it should be possible. Only when something has not been built well and looks badly will people notice and study the structure to figure out why it doesn't look right.

Designers often build levels with real-life sizes in mind: a door is X wide, an object is X tall, a wall is X thick, etc. Depending on the gameplay, field of view, and player movement, this may not actually work well, especially if the gameplay is fast and furious. In general, it's usually better to overscale a structure than to underscale it.

THEMATIC DETAIL

Details are part of geometry as well. They ground the identity of an environment and enhance the visuals. As game engines and hardware become increasingly more powerful, the needs and possibilities for detail are only going to rise. The theme and story must be reflected in the details. When creating an environment it's important to ask oneself the question: What kind of objects do the humans, or other creatures that live or work in this world, require? What tools, food, material, art, and so on would they need? A room can't be empty if the story just told the player four bandits live in the room. It should be clear that they do because of the details in the environment.

A frequent excuse to avoid adding details to an environment is that they will not be noticed while playing the game because of the speed of the gameplay. On the contrary, they will be noticed. There will be times when the players stand still and look around. In fact, items like details in the geometry might be the actual reason why they do. Additionally, no matter how fast the game is, details will always catch the eye of the player. One's eyes can identify an object in a ridiculously small amount of time. Feel free to test it out on your own.

TERRAIN AND NATURAL LANDSCAPES

TERRAIN

While most of the information ahead will focus on games and engines that use heightmap-based terrains, for example *Crysis* and *Unreal Tournament 3*, the information can be applied to modeled terrain (terrain created in a separate 3D modeling application and then imported into the engine), as well.

Compared to modeling terrain with brushes or a separate modeling application, heightmaps offer greater flexibility and speed at some cost to freedom. A heightmap terrain can be easily modified quickly and it is simple and efficient to texture. However, because of its base creation simplicity and the little effort required to create larger versions, it also means that mistakes are often made.

The downside of this type of terrain is that they can usually only control Z-axis (up or down) deformation. In other words, they cannot create overlapping features, like cliffs that push forward back over the terrain below. *Crysis* is an exception to that.

A heightmap terrain also always uses the exact same size for every polygon. Nature requires the exact opposite and thus heightmap based terrains can end up feeling somewhat unnatural and limited. Because of this, heightmap terrains should be reserved for smooth terrains, like rolling hills because anything more complicated might look unnatural.

In the event that terrain other than softly rolling hills is required, the sharper shapes should not be attempted by using the heightmap. In the following example the artist attempted to create a rougher, more random terrain, using a noise feature, which ended up looking ugly, pointy, and too evenly distributed.

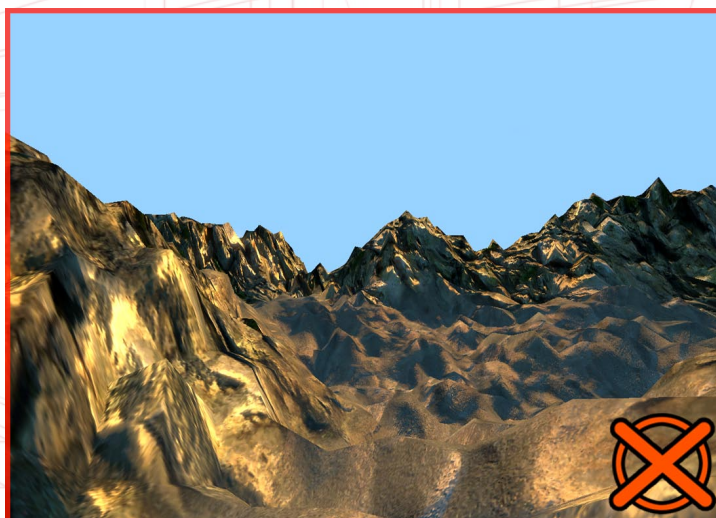


Image AV 28

The contrasts in height between nearby vertices are too large, which results in many unnatural-looking areas. The designer should attempt to hide the perfect grid of polygons that compose the changeable points of a heightmap terrain. Obvious sharp angles illustrate the low polycount of the terrain. This can break the illusion of the landscape. Smooth the terrain and hide the ugly edges. A good heightmap terrain is smooth everywhere with rocks made from separate meshes placed on top.

Creating a convincing rocky, angled environment with today's low and medium detailed heightmaps is practically impossible and will never rival modeled rocks in terms of quality. Also, steep surfaces are impossible with heightmaps that only operate on the Z axis.

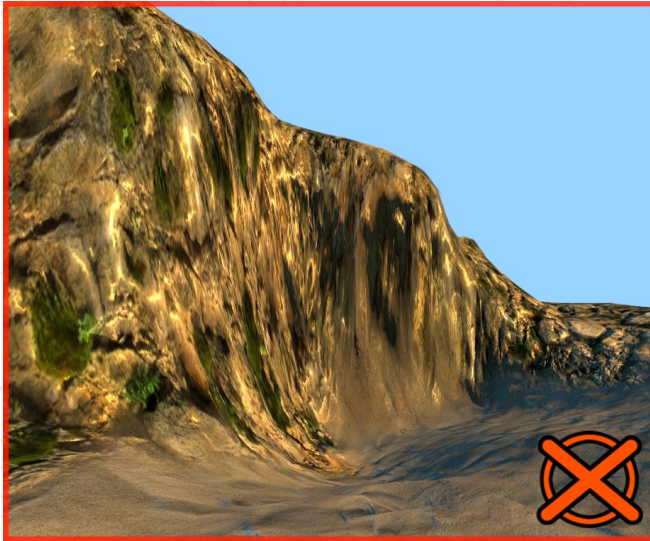


Image AV 29

If the level requires steep surfaces and rocky environments, the best option is to model those elements separately and add them as meshes.

In sum, heightmaps are best used to create smooth, flat surfaces such as grassy fields and sandy expanses. Rough terrain with cliffs and rocky outcrops are best left to meshes. These usually require sharp edges, random angles, and cannot be restricted to the Z axis. Rocks will be addressed in more detail after the following topic.

TERRAIN TEXTURING

The two most important aspects of terrain texturing is to create an interesting composition of different materials that reflect the theme of the level, and about adding enough variation to achieve the desired effect. Most of the content that will be covered in the Texture chapter also applies to terrain. Therefore, only the terrain-specific side of texturing will be discussed here.



Image AV 30

The textures used on a terrain should complement each other and the theme and form a coherent color palette. The example on the left is a case where this had not happened.

The terrain in the example mixes white snow with desert sand, green grass and a very brown rock. Where is the color and theme consistency among those choices? Stick to textures within a theme and with the same brightness and color qualities. This is a rather obvious example, so let's examine a more subtle example.

In order to achieve good terrain texturing it is important to not only choose the right textures and colors, but also to paint them correctly. In the following example the colors and choices do fit both each other and the theme of the level, but it still looks bad.

Due to the lack of variation, the texture tiling is quite clear which is not good. Painting too large an area with the same texture and without variation will increase the chance of seeing the tiling and it will be bland. As mentioned previously, a certain amount of variation is needed in order for something to be interesting.



Image AV 31

Try to break up areas by mixing some smaller areas of a different texture with the larger base texture. Bring in some extra variation by using more than one type of grass or rock, for example.



Image AV 32

In the left image, two different sand textures were used in order to break up the large open area. Variation is the key. Don't be afraid to add more textures than is absolutely necessary. I often use at least two sand textures to polish an area and add much needed variation. Surfaces such as sand aren't exactly the same everywhere on a terrain. For example, sand on a path looks different than sand under a tree, or in the middle of a desert, or on a beach. Paint those differences into the

terrain. It will not only make more visual sense but also provide the necessary variation. The same principle applies to anything on terrain: grass, snow, dirt, etc...

Yet another reason to use multiple variations of more than one type of texture is that it allows areas to be trimmed off – just like with more standard, structural geometry. Trimming always helps polish a level off and helps define area borders. Smooth the transitions with textures variation.

With terrain, trimming usually involves using a third texture to create a smoother transition between two different textures; the more variation the better for these situations. With only one sand texture and a single grass texture, the terrain in this example looks boring, simple, and awkward. By adding a second sand texture, it can be used trim off the primary sand texture and smooth the transition from sand to grass.

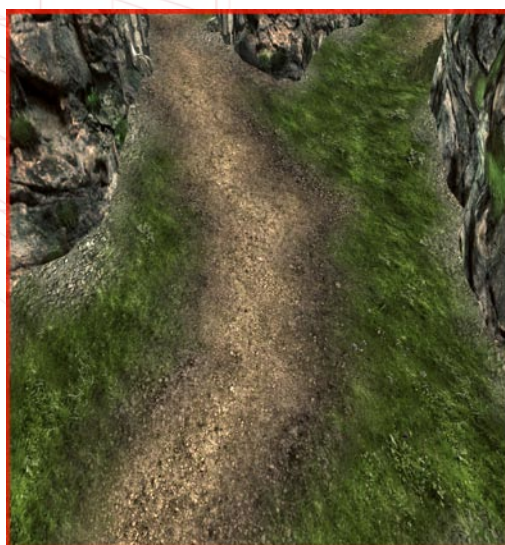


Image AV 33

In order to complete the locale's coherent look, it's also important to demonstrate how one element influences another. With terrain texturing this translates into painting the area around objects with a different texture. Thus the regular texture is broken up by a new one – the trim – and the object then shows a connection with the world around it. The object obtains a firmer connection to the area because it appears to influence it. If the terrain texturing remains unchanged next to the object, it will look more like the object has been dropped into place by the hand of (a) God rather than having been built there or grown there. The areas next to objects can and should be trimmed, as in the example below.



Image AV 34

The snow painted onto the terrain around the rock eases the transition from terrain to 3D model. Also note how the terrain itself makes a slight bump around the rock, to further ease the transition.

ROCKS

Again, it is much better to model rocks than to try to create them through the heightmap in a terrain. Heightmap-based terrain is simply too restrictive for rocks. Creating a good looking canyon with just a heightmap terrain is quite out of the question because it will result in ugly steep walls.

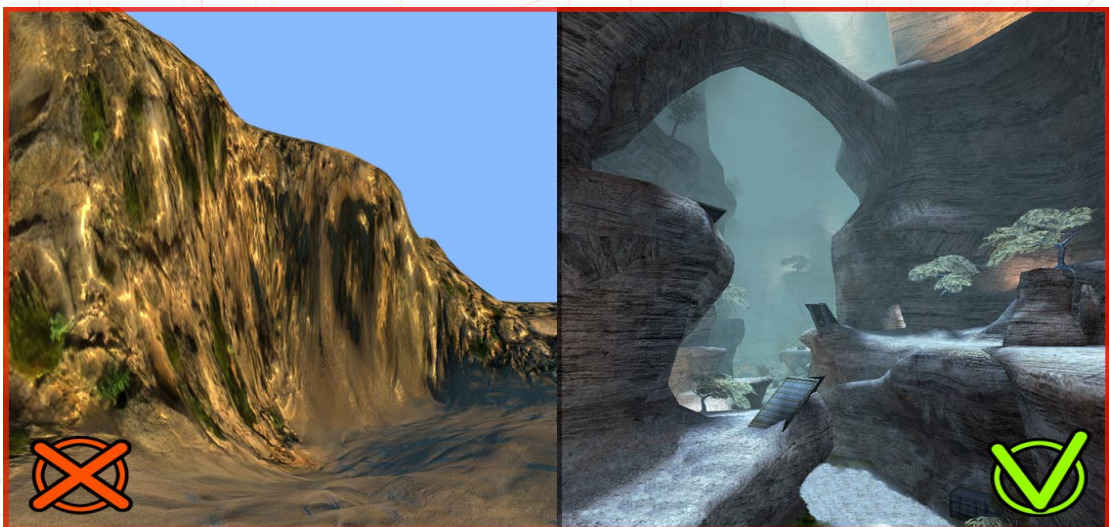


Image AV 35

As can be seen in the example on the previous page, modeling the walls will dramatically improve the looks of the canyon. In the right image, only the floor of the canyon uses a heightmap terrain for the base and the canyon walls are modeled elements. Combining models and heightmap terrain is strongly recommended for those features that are not smooth or low-profile.

In this example the heightmap is limited to the floor the players walk on while everything with a rock texture is a modeled asset. The terrain blends into modeled rocks near the edges.

Rocks can be constructed from dozens of styles and looks and it is exactly because of this that they are a great tool to bring more style to a map. They can have a strong influence on the visuals and since they are so flexible it is easy to modify them in such way that the map will benefit from their looks. The type of rock should reflect either the environment's theme or the atmosphere, or if possible both.

They should always reflect at least one of the two elements. Show the influence, and thus the connection, between the different elements.



Image AV 36



Image AV 37

In this example the rocks are round and smooth. It demonstrates how rocks can reflect the environment's theme. The rocks here are very smooth and round because of the erosion prevalent in the area. The environment in the screenshot was once below water. Also, this portion of the world was one of the more friendly areas in the game. "Round" is a friendly shape so it also reflects the atmosphere in this case.

I designed a different rock style for every world I worked on for this game in order to reflect both the theme as well as the atmosphere and to ensure each one had a unique look.

One of the other worlds in the game is an unexplored, dark, dangerous place.



Image AV 38

As can be seen, this one received a very spiky look. It is a good example of reflecting an environment's atmosphere and theme. The very spiky rocks look dangerous and scary, exactly the type of feeling the rest of the world required. Sharp is generally an aggressive and unfriendly shape.

Rocks are more than a collection of random shapes; the shape itself can have a significant influence on the feel of a level!

Many tutorials teach aspiring artists to create rocks and caverns by first creating basic spheres or cylinders and then adding a random noise effect. This method always results in a very random and generic look; the opposite of style.

Both examples at the top of the next page just don't reflect anything about the style of the environment they were made to fit into. By using the noise method, the resulting rocks have no identity and do not reflect anything.

The best result is always achieved with plenty of manual work instead of relying completely on such automated processes. No effect or filter can create something of the same level of quality as manual, purposeful work. The point of automated processes is to help the user with the creation process, not to create the actual product!

When making rocks for an environment one should ask certain key questions. Is the environment friendly or dangerous? What is the history of the place? How did nature form the rocks? Was it erosion from water or wind? Was it shaped artificially by an intelligent species? How will the other geometric/architectural elements look in the level? Will they all be cubic? If so, then the rocks should stick to that pre-existing style because consistency

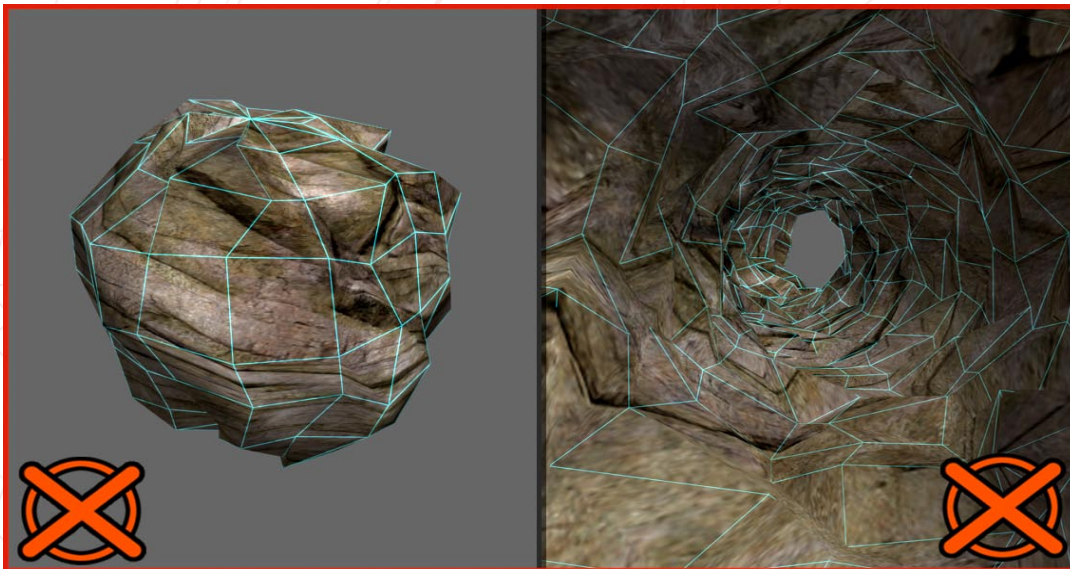


Image AV 39

between different elements is really just that important. If not, the rocks can be styled according to the atmosphere and history of the environment.

There are various modeling techniques used to model rocks. The technique to choose is a personal preference. I personally use box modeling most. Whatever technique is chosen, the original base model should never be able to be seen in the finished asset.

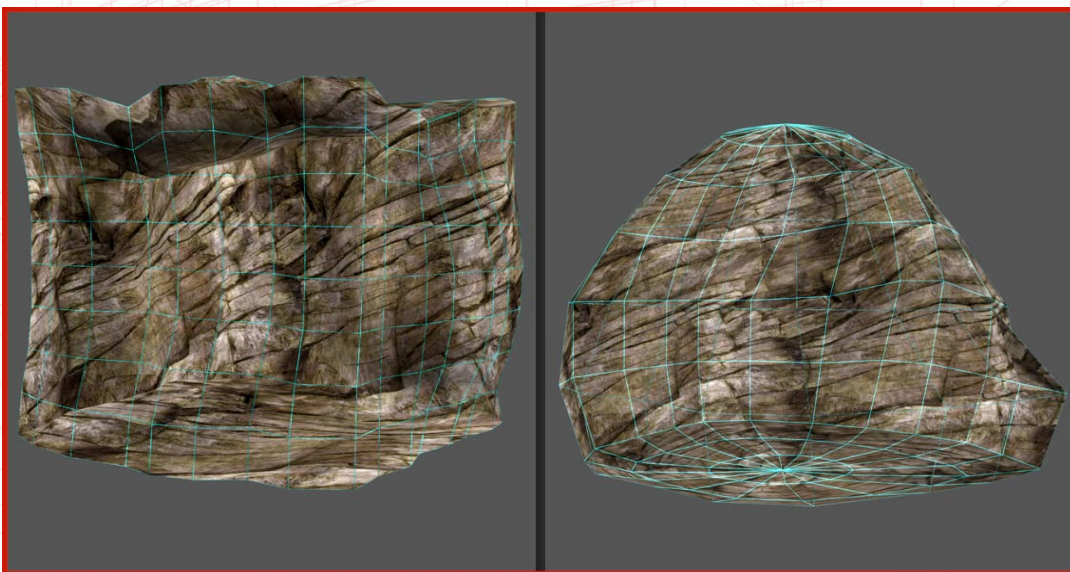


Image AV 40

Both of these examples illustrate how the original basic cube and sphere can still be recognized. This almost always results in weird and fake-looking rocks. Make sure the base shape is hidden well. Don't stick to the original tessellation; weld areas and create new cuts where necessary. Make sure the players can't see the perfect nature of the original tessellation grid. Make it imperfect and natural.

Anything that is either an obvious clone of another nearby element, too symmetrical, or forms a straight line with one or more objects will feel artificial. This is important in terms of both the modeling as well as the placement.



Image AV 41

The rocks in the left example are roughly ordered according to height. They also all have the same rotation, and a number of rocks have the exact same scale. The right example is made with the exact same model, but it was randomized more, and looks more natural. Those are the three things to look out for: the same scale, rotation, and wrong order.

HORIZON

As mentioned before it is crucial to give the player the feeling that the world is larger than just the playable area. The gameplay area shouldn't feel like the entire world rests just in that location. Horizons can help achieve the feeling that the world doesn't stop at the edge of the level. If the edge is defined by mountains, as in the following example, there should never be only one row of mountains surrounding the level. Or worse, it should not appear that the mountains are cut off as in this bad example.

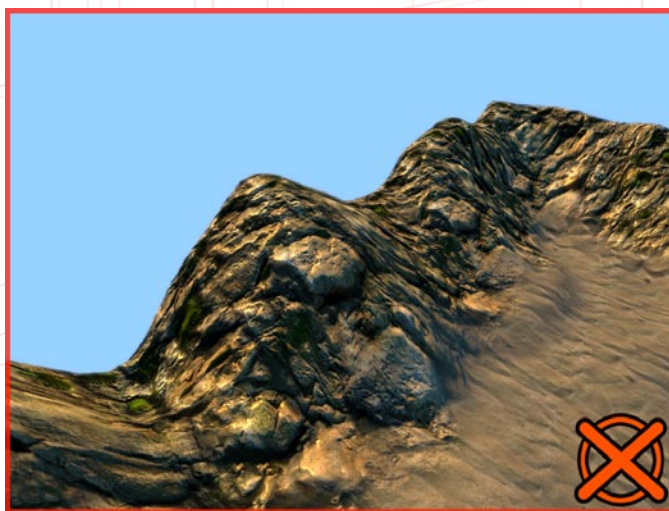


Image AV 42

player to the play area - make enough to help them imagine that the world goes on beyond them.

The mountains should continue on past the edge of the level. The mountains are not intended to be just a player boundary and have no other purpose than to simply block the player, they are also supposed to create a backdrop and a horizon for the world. The same goes for other large surrounding elements such as city buildings, forests, hills, and so on. If the intended impression is that there truly is a forest or a city surrounding the play area, then give the player the impression the area is much bigger than just the few trees around them. Do not make the minimum of what's required to restrict the

Another possible solution to the problem displayed in the image is to simply make the mountains of the world so large and high that there is no way the player can look over them. However, the experience this gives the player is rather uninteresting. Encircling a level with large walls will translate into a claustrophobic feeling. Instead, try to impress the player with open spaces and depth.

THE SKY

Closely related to horizons is the sky. Just like the horizon, the sky should also give the player the feeling that they are in a large and living world. Sky opens up spaces. The sky is one of the most important elements in outdoor levels for the simple fact that, depending on the game's camera type, it fills roughly half the screen most of the time. Therefore, the sky is a very important element that should offer the player more than just a handful of stars or an empty blue sky.

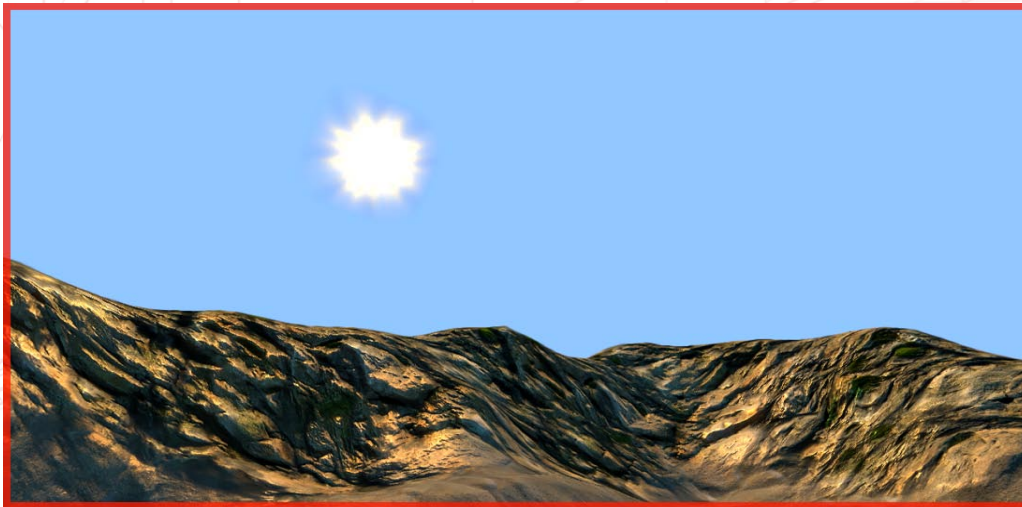


Image AV 43

This is a bad example because the sky is overly simplistic and boring. Skies can be very impressive, interesting, dynamic, complex, and beautiful. Use their strengths to reflect the atmosphere and theme of your level. It should fit the rest of the environment, form a unity, and actually add to it.



Image AV 44

This sky tells a story. It has drama and variation and it can reflect the level's theme and atmosphere. It is therefore a good sky.

The reason why a simple blue sky is boring is because it is too repetitive; just like any big flat surface. A sky needs composition just as much as anything else and for that it needs variation and contrast. A successful sky should be broken up by details, special features (the storm is one example), lighting, or even all of the above.

If it is decided that different elements are needed to break up a sky, ensure the disparate elements are coherent. The colors and elements must complement each other and be spread out somewhat evenly in order to form an interesting composition.

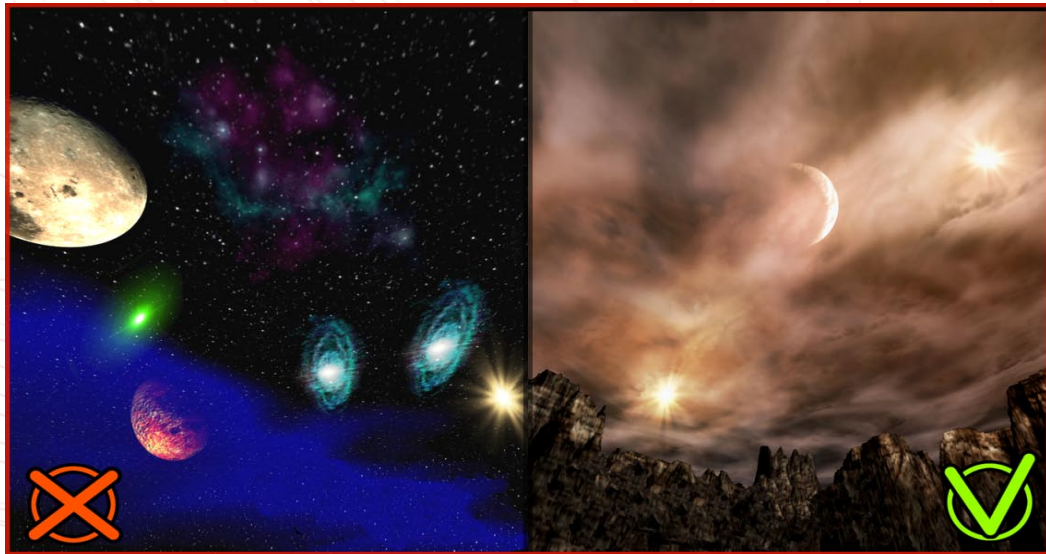


Image AV 45

The left image above is simply a mess. All the elements are scattered around randomly and their colors and shapes don't exactly complement each other, and therefore potentially the rest of the level. The right image is a much better example. Everything is spread out nicely, the elements complement each other, the colors are consistent and also are restricted to a small sample.

Another way to create interesting sky compositions is to use lighting; namely, colored lighting. Contrary to what many people think, skies can be lit as well – in fact they have to be. Whether the actual sky is lit, or whether the it requires painting the light into the texture in a 2D editing package, the results will be similar. Skies need to have light sourcing in order to illustrate where the light is coming from.



Image AV 46

The sky in the left image has no light sourcing. The sky in the right image does have light sourcing. While the light-sourced sky has a typical *Unreal* look, the point is that lighting within the sky does look more interesting. Lighting the sky will not only create a sky with more contrast and make it more interesting, but it will also make it more coherent. When one element of an area appears to influence another they therefore work together to create the illusion that they are actually in the same space.

A common mistake is to create a sky that doesn't fit the lighting or theme of the level. The sky is often the most important light source in the environment; especially if it is outdoors. Make sure the level communicates this. If the sky is very bright and orange the level therefore shouldn't be dark and blue.



Image AV 47

While this is an extreme example, certain designers do make this kind of mistake to a lesser extent. For example, using an orange evening sky but paired with an almost completely white lighting. Another example is placing the sun in the east, whereas the lighting comes from the south. Or simply having no apparent light source in the sky, yet a level that is obviously lit from the south.

The sky's lighting direction, color, and brightness must influence and match the level. If the sun is on the right, then the shadows in the level must match and extend to the left. If the level is a very cold and scary place the sky should not be a warm orange color with a happy sun; it should be dark. This is the reason why I personally would never add, for example, an orange sky to a snow level: orange is simply too warm a color. We will get back to this later.

Something else I consider a requirement as to skies is that they should be dynamic – they should move. It's not very difficult to add an extra layer of panning clouds to liven it up. One could also add some birds, airplanes, particle systems, moving light rays, and more. Levels are not supposed to be static pictures. One of the most important features of game worlds is that they are interactive and dynamic.

TEXTURING AND MATERIALS

INTRODUCTION

Textures and materials are level design tools that allow the walls to look like concrete and the floors like sand. Texturing communicates the type of environment to the player. It identifies the world and helps give it context – at times even more than geometry and sound. Textures can differentiate an old, raw, corroding cement ceiling from a well-built new one. Textures strongly influence other key aspects of the environment. Bad texturing can cause problems with lighting and can make the architecture appear artificial and out of place. Good texturing is essential for a good environment.

BASICS

Some of the most basic attributes of successful texturing can still go awry in the hands of beginners. Textures that don't match the theme and bad alignments, ensuring they physically fit together well, are simple aspects that many people consistently miss. Another basic rule is to scale the texture appropriately; to shrink or enlarge it properly.

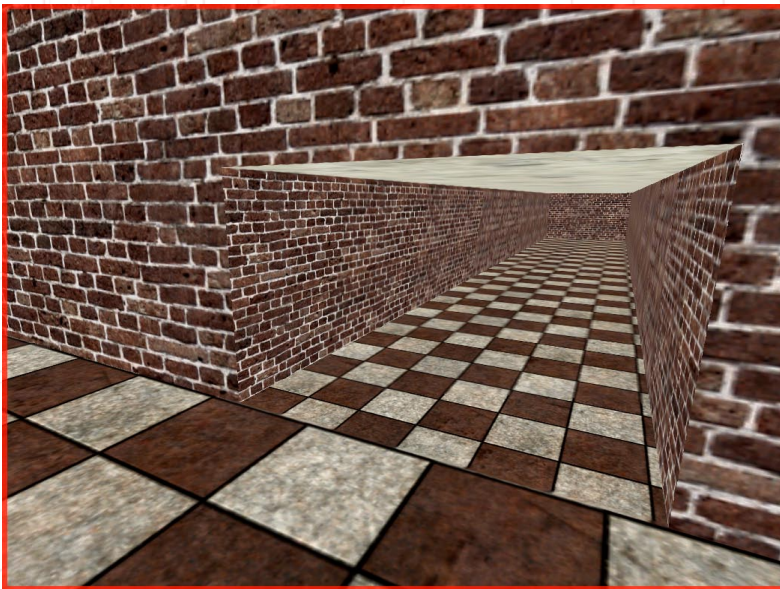


Image AV 48

In this example above, the textures are scaled randomly and inconsistently. The wall and floor textures has a smaller scale on some surfaces than on others for example.

Awkward alignment and scaling looks very messy and reduces the credibility of the world. It is very important to scale textures correctly. The player will not be immersed in the world if it is full of bad alignments and scaling. It tells the player that the world is artificial. As mentioned before, why would a player be scared of something that is clearly fake?

While this doesn't necessarily mean that hyper-realistic graphics and textures are required, nor that you may never scale a texture, it does mean that the world as a whole should appear contextual and 'realistic'. A player can become immersed in a well made cartoony world just as well as they can in a super-realistic game. However, this won't happen in either if there are badly aligned, stretched, over-scaled textures everywhere. They will simply break the illusion. As long as the textures are coherent, whether in a cartoon or not, they will respect 'realistic' visual expectations.

Bad scaling can not only make the world appear unbelievable, but can also make them blurry. Sharpness has been an issue with textures for quite a long time. The textures should all be similarly scaled and have roughly the same resolution or, even better, the same pixel/poly ratio. A player will notice the difference if the floor of the room uses a high resolution texture and the wall uses a low-resolution one. Equality between texture details is important.



Image AV 49

Bad scaling can also occur on a smaller and more subtle scale. The tiles pictured should match the floor. This can be obvious when modeling an environment, but it is not as obvious when working with BSP-based environments.

The textures used in the environment should always reflect the theme and style. While this sounds logical one often still encounters levels that contain two or more texture styles or themes as in this rather extreme example.

Too many disparate elements mixed into one scene is bound to end up looking messy; especially with no clear reason or logic behind the combinations. Remember, all the items should feel like they belong to the same environment. The players shouldn't begin questioning what they see – if they do, the level's credibility drops.

Choose a texture style/theme and stick to it. The choice made will have a significant effect on the look and feel of the level so choose wisely. For example, wood textures usually don't go well in spaceships. Choose a texture 'palette' to work with and restrict the choices to that selection.

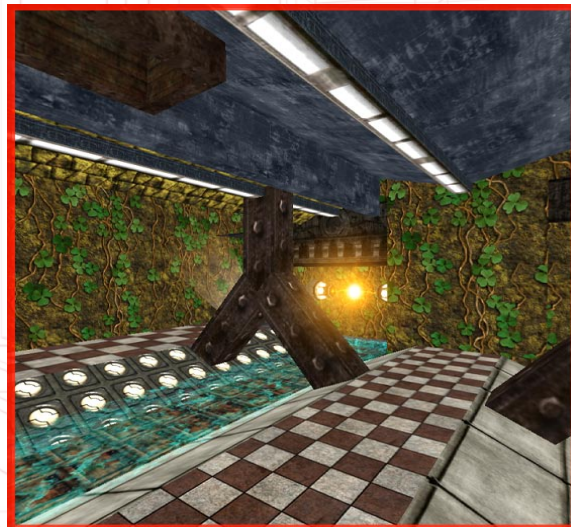


Image AV 50

In recent times, theme and colors mistakes are more likely to occur because of the increasing use of prefabs and similar content. All the different content may have been created with different styles and colors in mind. Thus, combining them may not be the best idea.

This also goes for reflecting the theme through the textures. If the environment is wet and decaying, then the textures to avoid would be clean and shiny ones.

Once I needed to create a swamp with ruins scattered throughout. However, the game's generic blue-grey ruins did not fit well with the green, wet environment.

In the real world the ruins would be much greener. They would reflect the environment they reside in by displaying more signs of age and decay. Instead they are very clean and the blue stands out too much. The ruins in this example screenshot simply are not consistent with the rest of the level.

If the environment is very dusty and dry then the textures should reflect this. For example, give them an orange overlay. Make them dirty and sandy. Doing so will not only improve the theme but will also help the overall composition in terms of color. And yes, nowadays color correction and post processing effects may also help, but you should never rely solely on those tools.



Image AV 51

As explained in the architecture and geometry chapter, trim is important. Trimming means adding a border around edges or a surface or texture. This prevents the texture from suddenly ending which is ugly and unrealistic.

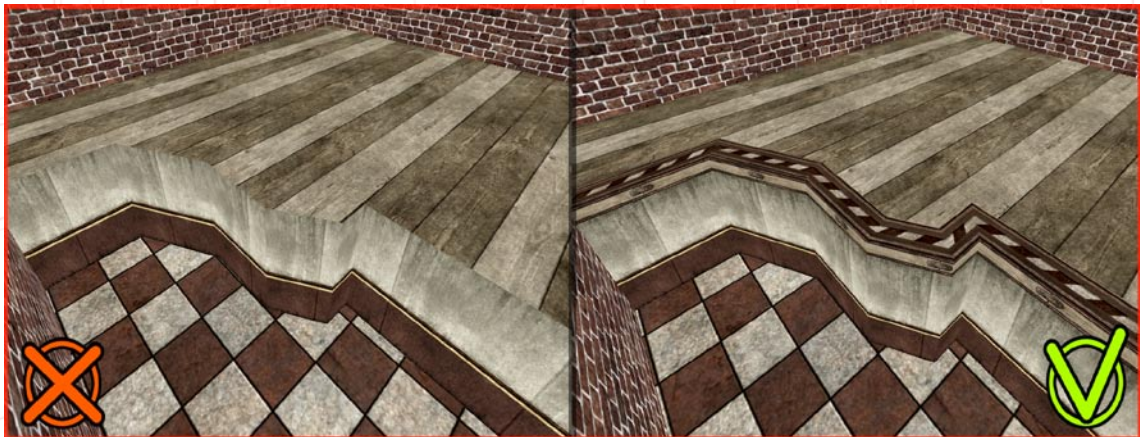


Image AV 52

While trim can never completely eliminate all the edges between textures it does reduce them. It also improves transitions between two very different textures. A nice metal bar between a wooden floor and a concrete floor can touch things up nicely and create a smoother transition between the wood and the concrete. It adds some extra variation and detail which is always good.

COLORS

Theme and style are not the only features to take into consideration when texturing an area. Color is also very important. Color can make or break the result. As will be examined in more depth in the Lighting chapter, colors can be used in many different ways. Texture color is very different from lighting color and also offers many extra possibilities. For example texture color, unlike lighting color, gives the option of using dark colors.

My advice for a good and consistent texture color palette would be to select a few nice subtle colors and stick to them for the textures.



Image AV 53



Image AV 54

In these examples the color palette for two texture packs I made can be clearly identified. The dark reds, whites, brown, and yellows all complement each other. They offer color variation while maintaining consistency and there is enough variation in the brightness of the colors to allow interesting compositions.

The colors chosen can be any mix as long as they suit the theme and offer some contrast without going too far. Colors in the chosen textures should form a balanced set, offer enough contrast to keep the viewer interested, yet also provide enough unity to remain consistent. In addition, they should reflect the atmosphere, theme, and style of the area.

Here are two other good examples.



Image AV 55

In the first unlit picture the dark red iron complements the yellow wood nicely and there are green plants around that offer a welcome variation on the dark red and yellow. There are only three main colors in the area. In the second unlit picture the white concrete and fabric contrasts the red bricks nicely, and all the steel is rusty which gives the steel a red overlay. All the colors feel coherent.

Both examples offer enough variation to keep a viewer interested yet they are also consistent. It is clear all the textures belong to the same environment - the colors and the texture choices fit together.

All the textures used in the environment should be restricted to a few colors. As shown before, a random mess of colors is rarely attractive, even if the textures do share the same theme and style.



Image AV 56

While all the textures in this example do belong to the same semi-realistic industrial theme and style they are, when assembled like this, a chaotic mess of different colors. There is no logic or system behind the choices.

The colors chosen are dependent on both the theme and the desired color palette. Adhere to a certain number of colors in order to create a unity that fits the theme.

One example of this is an autumn farm environment I once had to create for a project. I ran into trouble when I wanted to place corn on my fields because the corn was an asset from a summer level and thus looked too green to fit the brown, autumn color palette. I fixed this by duplicating the texture and re-coloring it. The green corn not only looked out of place in the autumn theme but it also didn't fit with all the autumn colors. Both the theme and the chosen color palette are important.

The same level presented another, less obvious, problem. There was a group of farmhouses on the land. The farmhouses were generic farms and had been used throughout a wide range of different levels. However, the problem was that they were quite bright and had a blue overlay while my environment was very brown and orange. They were too bright to blend in well. I ended up changing the colors of a few of their textures to bring them in line with the color palette of the level. This can be seen in the two screenshots on the next page.



Image AV 57

In modern games Post Processing can solve this problem to a certain extent. Post Processing allows the artist to add a color overlay over the entire level, similar to the color overlays commonly seen in movies. They can bring all the colors closer together which also results in a more consistent look. Post Processing or not, always attempt to stick to a coherent set of colors in the textures anyways, as one shouldn't solely rely on Post Processing to clean up whatever mess was made.

As mentioned before, using subtle colors is the best course of action because of the lighting. Textures are the base for the lighting. Bad texturing can destroy the lighting! The more saturation a surface has, or the darker it is, the harder it will be to light. Since lighting is almost never white, its color mixes with the texture color which can create undesired results.

It is much easier to draw with colored pencils on white paper than, for example, dark red paper. Texturing works the same way. When picking the texture color palette one needs to plan what light colors will be used and then adjust the texture choices and saturation level with the lighting in mind. For example: it is logical that yellow lighting works much better on yellow textures than on blue textures. Therefore if the majority of the lights are going to be a blue color then one should avoid using a lot of orange textures. The blue light will blend in with the orange textures and that usually looks strange

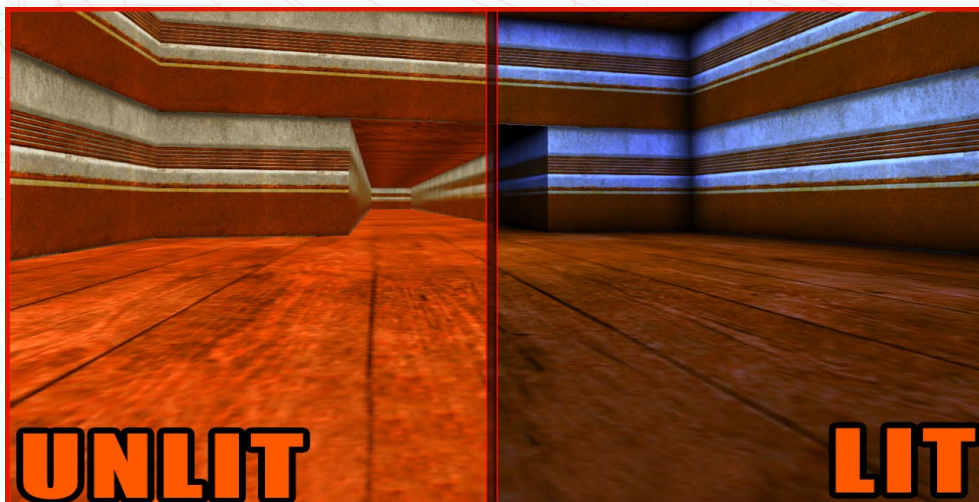


Image AV 58

As can be seen, most of the surfaces will not look blue even if there is a strong blue light nearby. If the textures are too saturated then the lighting will never be able to color it correctly, and the end result will be a weird blend of all kinds of colors; in this case a brown-red mix.

While color has a significant influence on lighting, the brightness of a texture could be said to have an even greater one. As the brightness and whiteness of a texture increases so does the ease at which it can be lit. A black surface, on the other hand, can't be lit. If the textures chosen are too dark then the area will be very difficult to light.

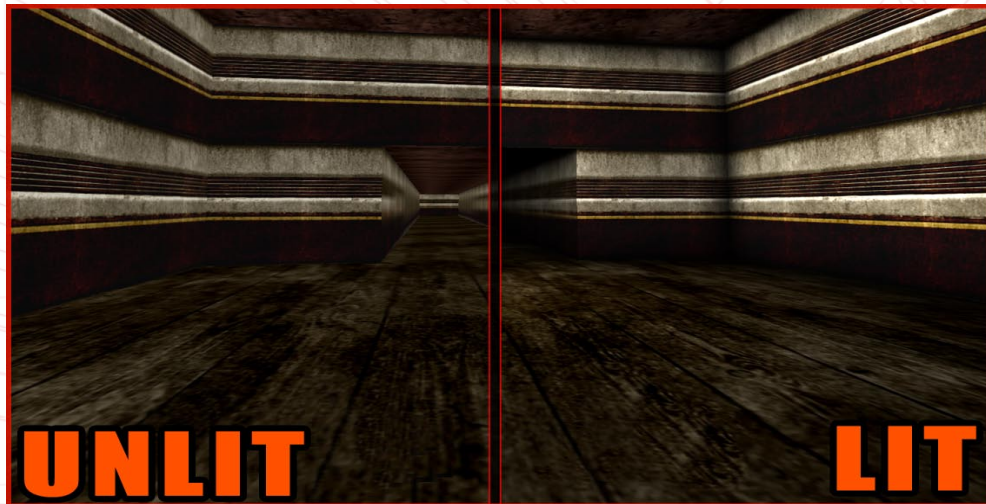


Image AV 59

The difference between lit and unlit can be minimal. The light in the lit picture above is unable to light the room very well at all because the majority of the textures are dark. Only the bright areas lighten at all and the darker areas show barely any difference between the two versions.

It's better to err on the side of too bright when making textures. They can still be made to look dark by lowering the light's intensity whereas this cannot be done with dark textures. All the textures should also adhere to a similar level of brightness. Using too many textures with widely varying brightnesses will cause the end result to look messy and balancing the lighting within the level will be difficult.

This balance and brightness issue is a common problem when working with snowy environments. Snow is white and bright and causes every other texture to appear dark. When working with snow, the other textures should be brighter than usual. You should be extra careful using too dark textures in combination with snow. If the lighting is strong enough to light up dark textures, then the snow will become overlit and blinding.

Some people sometimes argue that in real life there are many black and dark surfaces that look good and therefore there shouldn't be a problem with using them in games. As will be explained in more depth in the lighting chapter, games are still too simple. Lighting and materials don't act realistically. Some simple or dark surfaces work well in real life because of certain attributes that games still cannot replicate. Specularity, reflection, sub-surface scattering, and other properties can turn dark textures into interesting, viable materials. It's not the texture itself that is the issue, but its interaction with light - its reflective and absorptive properties. Unless the game engine used has an advanced material renderer then working with dark and black surfaces should be done with care.

MATERIALS

With ever-growing hardware power, designers and artists have ever-increasing control over materials. This control allows better manipulation of the environment than ever before. Each material can look very different. One material may be very shiny while another can be dull. As an artist, materials should be made to appear as close to the real world as much as possible.

But it can, and should, go even deeper. Materials should be used to set the mood of the environment. Shiny and reflective materials are commonly associated with 'cold', whereas dull materials are generally 'warm'. Water, ice, cold metal, and the like are usually reflective. Warm materials, like wood and hot metal are typically dull. If a shiny metal heats up it loses some of its reflective properties. Of course there are plenty of exceptions to this but the point is that the average person will unconsciously interpret shiny materials as colder than dull materials. Artists can exploit this to better leverage the feel of the environment. Similarly, shiny materials are commonly associated with 'scary' and 'sci-fi', to name a few. Dull materials are friendlier and better suited to environments that feel 'safe'. A scary sci-fi environment, like in *Aliens*, would therefore be best suited to shiny materials. A children's game, on the other hand, would probably be more suited to duller materials. Understanding this can benefit a player's unconscious interpretation of the environment and enhance the consistency within it.

It is also important to create an interesting composition using different material types. Spreading different types out across an area helps it appear balanced. If all the shiny elements are on one side, and all the dull ones on the other, then it will appear unbalanced. Lighting could restore the balance if strong lighting is used on the dull side and weak lighting used on the shiny side. But whatever the situation, always try to keep the balance instead of trying to fix it through lighting.

LIGHTING

INTRODUCTION

Lighting is one of the most important and influential elements in environments. It has the power to make or break the visuals, theme, and atmosphere. Even the most simplistic level can still be brought up to a high level of quality with nothing more than lighting. Likewise, even the most awesome architecture can look horrible if it is badly lit.

Lighting is often forgotten and underestimated. Designers often add it quickly and without much love. While in the past this was partially excusable by the weak hardware and game engines, these excuses just don't hold up anymore. Lighting is just as important as geometry. Without lighting there is no environment but just groups of three-dimensional objects. Lighting has the power to communicate the atmosphere to the player; for example scary or happy. Lighting has the capacity to bring life to a group of objects and bring them to the next level of quality. Its purpose has expanded further than just giving the players the ability to see where they are going. It creates and enhances the ambience. Lighting can make places look scary/cozy or cold/warm. It augments the three-dimensional feel of objects, creates composition and balance to lead the player's eyes around.

Yet, despite all of this, there are still a very large group of games and levels out there which use nothing more than white ambient light everywhere.

THE SOURCE

The most basic rule of lighting is that it always needs a light source. It is impossible to have lighting in an area with no source, as in this bad example.

While there is plenty of lighting in this corridor it's impossible to tell where the light is coming from. This completely breaks the illusion and looks artificial.

The opposite can also be true and is also bad. There could be light source in a level but no light coming from it, which is the second rule. A common mistake is to add lava in an environment without adding any lights nearby. Lava is molten rock and fire, both give off light. Thus lava should emit red or orange lighting. This is also true for any other type of light source such as a regular fire, the vehicle headlights, radioactive slime, and so on. An example of this can be seen at the top of the next page.

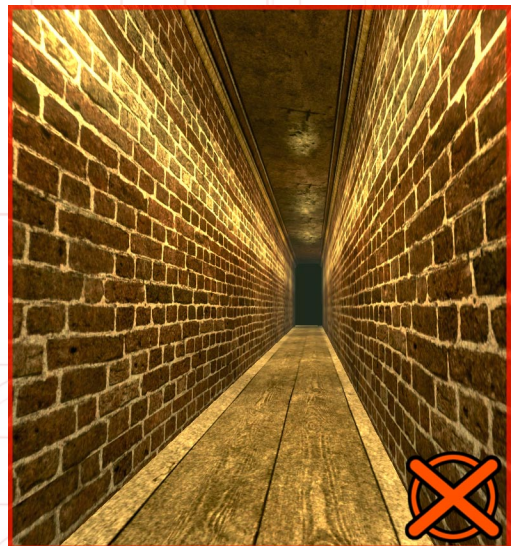


Image AV 60

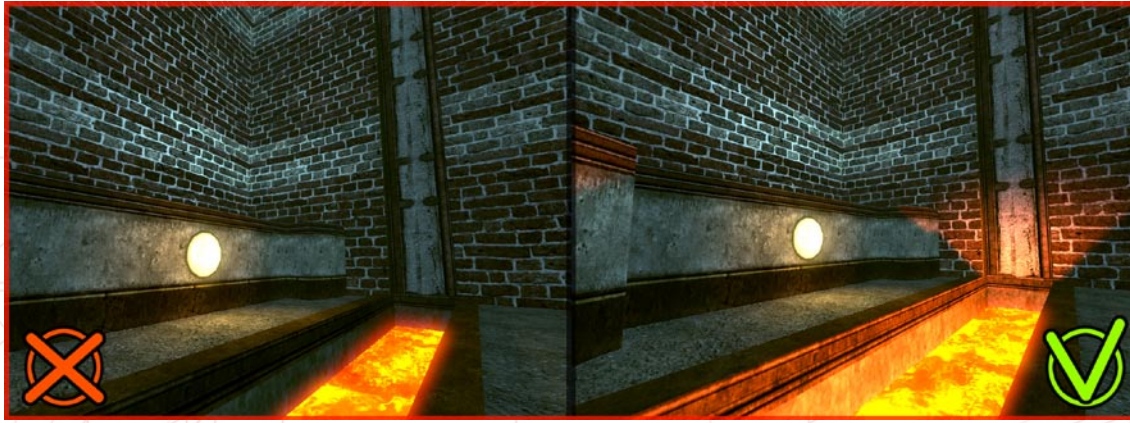


Image AV 61

Another lighting aspect to avoid is lighting that is out of balance with the size of the source. An example could be a small light source that somehow manages to illuminate an entire room or corridor, as in this bad example on the right.

Keep lighting in proportion to its source and its environment!

Light sources can be anything: small or large lamps hanging on walls or from ceilings, the moon or the sun, crystals,

lasers and other type of high-tech beams, fire, mirrors, magical effects, water surfaces that reflect light, lava, radioactive slime, and so on. Almost anything is possible as long as there is a noticeable source that is large enough to warrant the amount of light in an area.



Image AV 62

The same goes for the brightness of the source itself. If the lighting is very bright the source itself should not be dim. It should be just as bright and, if possible, be paired with effects, like a surrounding glow, to enhance the brightness.

The example on the left is bad because the lamp appears to be 'off' even though the environment does seem to be receiving a lot of light emitting from it. The brightness of the light source and the brightness of the lighting in an area must be balanced and appear equal.



Image AV 63

Similarly, the player should see exactly where the light is coming from. The area near a source should appear the brightest.



Image AV 64

The left image is bad because the entire corridor has an equal brightness, which is strange. It doesn't feel like the light is really coming from the lamps. The light should be considerably brighter near the source rather than ten meters further away in a corner. Its brightness should fade as it travels further and further away from the source. The light should show variation in brightness as it travels. This is not only more realistic but it also helps the lighting composition. Show a direct influence on one element from another!

If there are multiple identical light sources in a row, such as the lamps in a corridor above, it can help to give some of the lights a slightly different color or brightness to add additional variation.

LIGHTING COMPOSITION

Think about the concepts explained in the composition and architecture chapters. If there are no, or barely any, changes and contrast in an area, then the scanline chart would be a very flat line. No change is repetitive and is thus predictable and boring.

Lighting composition examines the way highlights and shadows are scattered around. If an area has flat lighting without any contrast, it will look boring. The best example of this is a corridor. By creating darker areas in between the highlighted areas the area gains some variation and thus it will be more visually appealing. Have a look at the example screenshot at the top of this page again.

This is also true for rooms. If the entire room only has one light source then there is a good chance the lighting composition is not being exploited to its full potential; which is especially true for highlights. Shadow composition is much easier to achieve because it is simpler to create. A single light can create quite a lot of interesting shadows by itself but it will be much more difficult to create an interesting highlight composition, although it can be done.

This is especially a problem for outdoor environments. Outdoor areas typically only have a single light source like the sun or the moon, and while a single light source in the sky can create nice shadows, it won't succeed in creating a nice highlight composition. Both are needed to create a good composition. The shadows need to play with the bright highlights and in outdoor lighting this is too often forgotten.

When I have to light an outdoor environment I always add highlights like small lamps, car headlights, torches, and any other small light sources that could give off light throughout the

area; especially in areas that are already in shadow from the sun/moon. Adding highlights outside the shadows would make them appear unbalanced as they would look much too strong. By adding lights inside darker areas, I blur the contrast between darker and brighter areas, and make the darker areas catch more attention.

This also depends on the theme. If the level has a nighttime setting, the environment will be dark enough to support highlights almost everywhere. Nighttime environments need highlights even more than daytime settings. There is nothing worse than a night environment that relies purely on ambient light or, in best case scenarios, just the moon. Due to the complete lack of composition, combined with darkness, the result is often worse than a daytime level that doesn't have a nice lighting composition. This is because daytime levels can still fall back on shadows and the colors from the textures to form a composition which keeps the player's eyes interested. In a night setting however this is not possible. The nonexistent, or weak, shadows prevent the level from having a shadow composition and the nighttime ambient light is usually either too dark or saturated to bring out the texture colors. The result is that the only hope lies in lighting highlights.

BLACK AND WHITE

Pitch black areas are dangerous and should be used sparingly and with caution. With the exception of certain themes and styles that rely heavily on black it is, in most situations, just unwise to use. While black is less risky than white, as it grabs less attention and is better in supporting atmosphere, it is still a very dead color. It is, in fact, the deadest color there is as it emits absolutely no life or atmosphere, and, by default, looks very uninteresting.

Large black spots make a level look unfinished, simple, dead, and artificial in almost all situations.

This example screenshot is really boring to look at because of the large amount of black. It also has a very disturbing harsh contrast. Pitch black shadows are unrealistic as there is almost always some light scattered throughout the room lighting up corners and other places. I always use subtle ambient lighting to get rid of pitch black areas, and make the level look less harsh and contrasting. Soften it out, but not too much!



Image AV 65

The same can be said for white; pure white lighting is almost always unnatural and usually ugly. White is a simple, default color. It offers very few possibilities and adds little emotion or atmosphere to most scenes. The color is neither cold nor warm and it doesn't create good contrast with any color, aside from black, which, as already discussed, should be avoided. Treat white cautiously and never use only white light in an environment except for very specific situations.

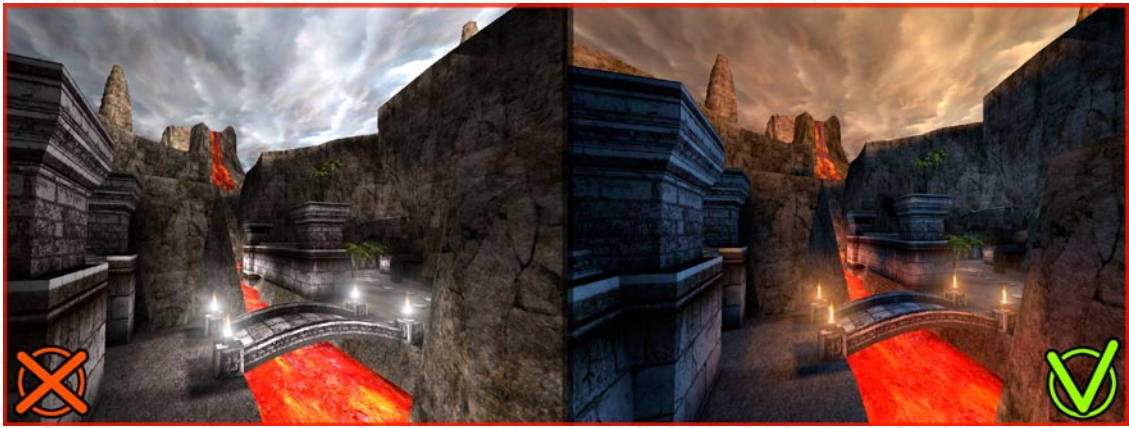


Image AV 66

White is also often overused because it is the default light color. It will therefore give the level a very basic and unfinished look. The resulting impression will be that the author was too lazy to put work into the lighting, even if they actually did. Color the lights, even if only very subtly. There should always be some sort of color added to the light. Even in the real world no light is truly white. Mix colors to create an interesting composition and to reflect the theme and atmosphere.

COLORS

One of the most complex aspects of lighting is color. Colors can make or break a composition as they shape the atmosphere and emotions associated with an area. They simply make environments more interesting and lively to look at.

Most light sources in the world cast light that, in one way or another, has color. Therefore it is not very realistic to place white lighting in the environment. For example, a lamp might cast a yellow light because it is enclosed in yellow glass. Or perhaps it is an old lamp and the glass has begun to change color due to the wet environment it is in. Or perhaps the light is shining on a yellow wall which causes the light rays to bounce off and carry the yellow color to another surface which results in seemingly yellow lighting.

Bounce light is known as radiosity and, up to now, there still aren't any games around that offer correct and complex radiosity lighting, and certainly not in real time. Therefore, until there is such technology available, one must color the light oneself, and basically fake radiosity lighting. If color isn't added manually, the result will be very bland and artificial.

Another reason to use colors is the composition. Using just one color is repetitive and boring; at least two colors are needed to create a contrast. If only one lighting color is used, that very important color contrast is lost and the result ends up very bland. This can be seen in the example screenshot on the next page.

Change is also necessary in order to form a composition and one color can not offer the necessary changes. The colors used need to strike the right balance between providing enough contrast yet still complement each other. Harmony is the word to remember well when dealing with lighting. One cannot mix just any two colors together.



Image AV 67

Before being able to work with lighting colors one must understand how colors work. There is a huge difference between the regular colors used to create textures and the colors used to light an area. Lighting is composed of Red Green and Blue (RGB). CRT monitors and TV's use this system as well.

On the other hand paintings, pencils, prints, and so on are Cyan Magenta and Yellow (CMY). They operate on three totally different primary base colors. In very simplified terms they are blue, red, and yellow; the primary colors - which are taught when one is very young.

The difference between the two systems lies in how they create colors, or how they mix. CMY colors will end up as a brown black mess when mixed together. Think about what happens to all the colors when all the colors in a paint palette are mixed together.

RGB, on the other hand, will end up white when mixed. Shine multiple colored lights at one spot and they will end up creating a white spot.

The important difference for is that certain color combinations which work great in a painting will never work out when lighting game spaces, and the other way around!

It is impossible to use a color combination that works in CMY for lighting because of two reasons.

First off, every color is a mix of the three primary colors. RGB mixes differently so the colors it creates will look different than those created by CMY. This is especially a problem when one color accidentally mixes with another in spots, something that is bound to happen when working with lighting. Mixing blue and red in CMY might look nice in paint but when red and blue are used in lighting they will create bright purple spots. Certain variations and mixes of colors are not the same in both types.

Secondly, RGB simply doesn't have all the colors CMY has, lighting-wise. Therefore, converting color combinations is not always possible. RGB does not have dark yellow, dark red and so on. It can't create dark colors because lighting cannot be dark - it is always light. It can, however, be more or less saturated, or more or lesser intense, but not dark. Black light does not exist and neither does gray or brown light.

One could say that lighting uses simpler colors and has more limitations. There are fewer colors and less subtle changes available because of the lack of dark colors. Lighting is constrained to a relatively small set of colors.

What makes this even more difficult is that half of the available colors almost never function well in most themes and subtle changes in hue or saturation are barely noticeable. Colors like purple or pink are almost impossible to use in most themes and styles because they simply do not fit, nor do they look natural. Using them will most likely result in some weird and unrealistic neon/disco style. The palette of useful colors is very small and mainly consists of yellow, orange, blue, cyan, red, and some green.

Never use traditional painting logic to rely on mixing brighter lights with darker lights to create changes in the environment. Dark colors do not exist! A difference can only be seen when a light's color or saturation radically changes, so subtle changes won't be noticeable. This is because light is always a gradient. It always creates a lighter area and a darker area. Lighting simply starts somewhere and then fades out as it travels further away from its source. If one attempts to create contrast by using darker and brighter lights of the same color, then the result will look weird. The lights appear too weak to be realistic, and it will feel as if there are too few lights around.

Now that the theory of color has been explained, it is time to apply this knowledge to lights in a level. The idea behind colors is to add to the theme and atmosphere and to help create a composition to aid the eyes and to keep things interesting.

COLORS CONTINUED: COMPOSITION AND CHOICE

Colors create various contrasts and feelings. It is essential to understand them and use them correctly in order to create interesting and appropriate lighting.

As mentioned before, the key to creating an interesting look and composition is to use well-balanced contrasts. Too little or too much contrast is bad.



Image AV 68

Neither of the images above look good. The first picture is very repetitive and thus boring because everything has the same light color. The second picture has so many different light colors that there's no harmony and it looks completely random. This is also undesirable.

Avoid weak compositions or very harsh ones. When converted into the scanline charts, the above two pictures show clear problems.

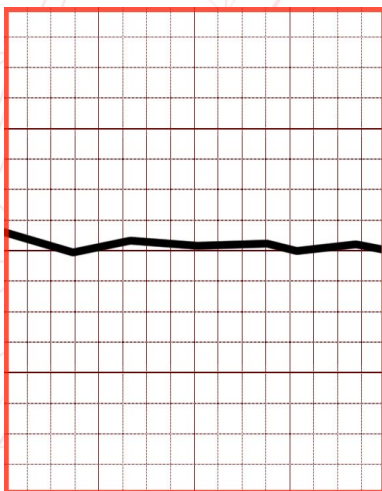


Image AV 69

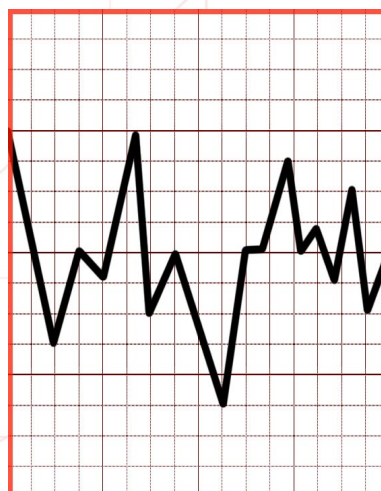


Image AV 70

The line either has very little change or the change is so hard and sudden that it is not pleasant to look at.

It is for this reason that the right combinations and placement of light colors is needed. I personally always use two primary light colors such as blue and yellow and then a third color, like orange, to give extra contrast and accentuate a few special elements.

The third color is to prevent the two main colors from becoming repetitive. Too much use of the same combination can also become boring. The third color's purpose is to occasionally break up that combination.

I usually refrain from using four colors because too many colors can make things look random. It should never look like a mess; unity is the goal.

Composition-wise, light colors should follow the same rules as highlights. Their composition should be evenly spread out so that there are no large areas of the same color which would unbalance the visuals in an area. If the entire right side of a room only has blue lights and the left side has a mix of blue and yellow lights it might appear unbalanced. However, this is also dependant on the composition of other elements such as the architecture or any moving geometry. Having strong lights on one side and a moving piece of geometry on the other might balance it out.

Now, one may wonder what colors to use and combine. Combining colors in lighting is about more than just finding a random combination that looks cool. There are systems and arguments that can help create the right combination. The light colors should not only enhance the visuals and the composition but they should also enhance the theme and atmosphere. The choice of which colors to use depends, for a large part, on the theme and the desired atmosphere. A scary theme would, for example, require cold colors.

There are different types of color combinations and each one offers a different type of contrast.

First, there are the cold and warm colors. Some colors feel cold, such as blue, while others feel warm, like orange. Cold colors are blue, green, and purple. Warm colors are yellow, red, and orange. It is logical that combining a warm and a cold color can produce some nice results.

Another type is the strong color combination. Some colors are very aggressive and powerful while others are very passive and relaxing. Strong colors grab attention even when used in small amounts. Red is the best example of this. It is such a powerful color, that even a small spot of red in an environment becomes dominating.



Image AV 71

In this picture the one thing that stands out the most is the red light. With its small radius and low brightness, it even manages to beat the very strong white light in this example. Red is incredibly aggressive and thus should be used with caution since it can make the player forget everything else in the scene, which may not be the intended effect.

Other aggressive colors are, in order of aggression, orange and then yellow. Relaxed colors are those that invoke comfort and calm. They rest the eyes. Relaxing colors are blue, green, and purple.

The last type of color combination is light and dark. Is the color closer to white or closer to black? The simplest way of checking if a color is light or dark is to think about what would happen if it was converted to grayscale. What would a photocopy machine do to it? Red is a dark color. It becomes almost black when converted to grayscale. The same is true for blue and purple. On the other hand green, and especially yellow, are bright colors.

Choosing the right color is not done on a whim. The better the choice, the better the contrast will be and therefore the better it will look. For example, two cold colors should not be chosen as the primary light colors. One is better off combining different types of colors together like a warm orange with a cold blue.

The best light color combination to use is yellow and blue, along with all similar variations, such as orange-blue and yellow-turquoise. Yellow is a bright, aggressive, warm color while blue is a dark, relaxed, cold color. It is the only combination that manages to use the opposites of all three types, which is also the reason why it is used in so many games. Yellow is also subtle enough not to draw all the attention to itself like red would. And, additionally, it is also the most natural combination; more on that later.

To complicate lighting even more, there is one element which can make the effects of each of these different types either stronger or weaker and that is saturation. White is a special color that feels neither cold nor warm, neither aggressive nor relaxed. Aside from being a bright color, it is the epitome of neutral. Therefore, lowering the saturation of a color can neutralize its effects a little, which can be useful. In order to achieve a balanced look it's necessary to find the right saturation for the colors. If all the colors are one hundred percent

saturated the result would be a very harsh look with very strong colors.

While creating contrast, unity should not be forgotten. Find the balance between the two extremes. In the example on the right the contrast is too harsh resulting in an ugly, unbalanced, and unrealistic scene. The balance between the two colors forms the key to success. I usually pick colors that are only fifty percent saturated but whatever works for the particular situation is the best choice to use.

Slightly desaturating the primary colors is advised, although there are always exceptions. For example, colors like red will turn pink when desaturated. There are also a few light sources that always need very saturated light; fire for example.

The amount of saturation can greatly alter its look or feel. A very white blue feels colder than a very saturated blue. This is important when trying to achieve a cold feel.

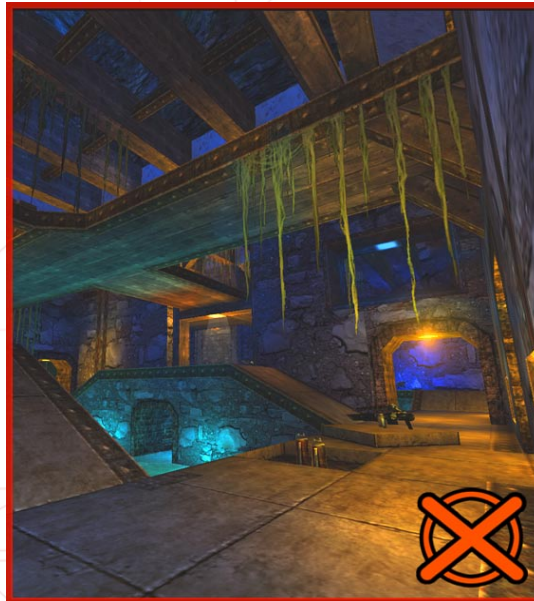


Image AV 72

This brings us to another very important point: theme. As mentioned before, colors are not randomly chosen. 'Because it looks nice' should never be the sole reason why color X is being used. The color combination should not only fit together but it should also enhance the theme and atmosphere. For example, if the theme is an ice environment, then many warm colors, like orange, should not be used.

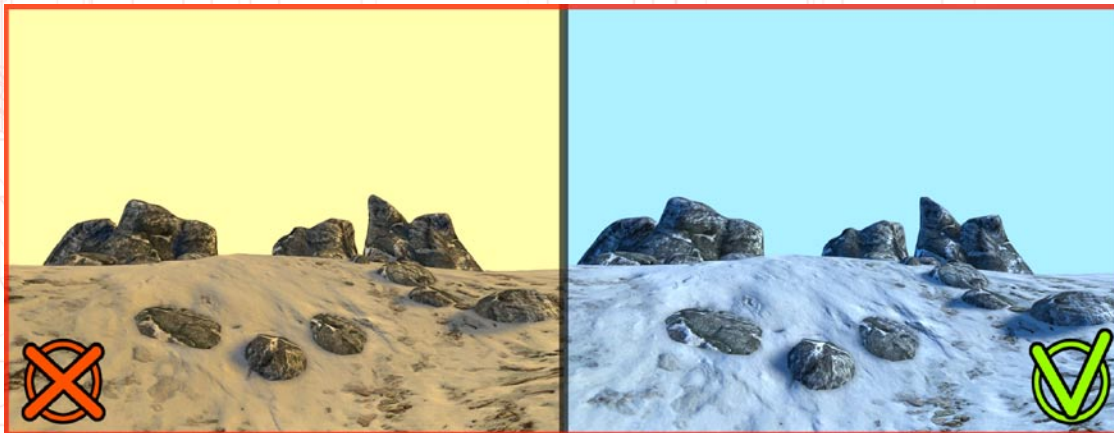


Image AV 73

The left example is bad because warm lighting is being mixed with a cold environment theme. The example on the right is good as it feels colder. A cold environment needs cold colors like blue.

People associate colors with feelings. The whiter the color is the cleaner or colder the area will appear. On the other hand, darkness is experienced as scary or depressing. When I design a new level I always ask myself the question 'What color do people associate with the theme I have in mind?' If I design a lava environment it's very clear I will need a lot of red and orange lighting.

After I have my first primary color I try to find the second main color. The second primary color has to create a contrast yet look nice in combination with the first color. If the theme involves lots of water or a sea then my first primary color will be blue and my second will be yellow. The lava example would have red and orange as the primary color and blue as the

secondary color. A dawn environment would usually have light yellow as the first main color and light blue as the second primary color. Humid environments feel better with some green and a very dead environment might do well with subtle red or purple lighting.

As mentioned in other chapters it is, in the end, all about clichés. People need to quickly recognize something and the easiest way to do so is to communicate through clichés.

Sunlight is perhaps the best example of how contrasting colors work, especially in the evening. Unless it is noon, direct sunlight is always slightly colored. Think about what color the sun has in the evening or at dawn. It will appear as orange or yellow most of the times. Indirect sunlight has a color as well. It is usually a blue or slightly purple color, where the purple comes from the mix of blue and orange.



Image AV 74

In these evening beach photos the color of the sun and ambient lighting is readily apparent. The direct sunlight is orange while the ambient light is blue.

White lighting is, in almost all situations, unrealistic; just as coloring an entire outdoor area with the same color would be. In most situations there should always be two colors: one for the direct sunlight, which is likely a type of yellow, and one for the indirect sunlight, which is usually a type of blue. Not only is this more realistic but it will also look better.

TEXTURING AND LIGHTING

Texturing can make or break a level's lighting. Textures are the base for the lighting. The texturing of the world carries a large amount of responsibility. While I already touched upon this in the texture chapter I would like to give a few extra examples of common mistakes.

If a texture is too dark it cannot be lit well. The same goes for too bright or white textures. They will look overly bright when lit.

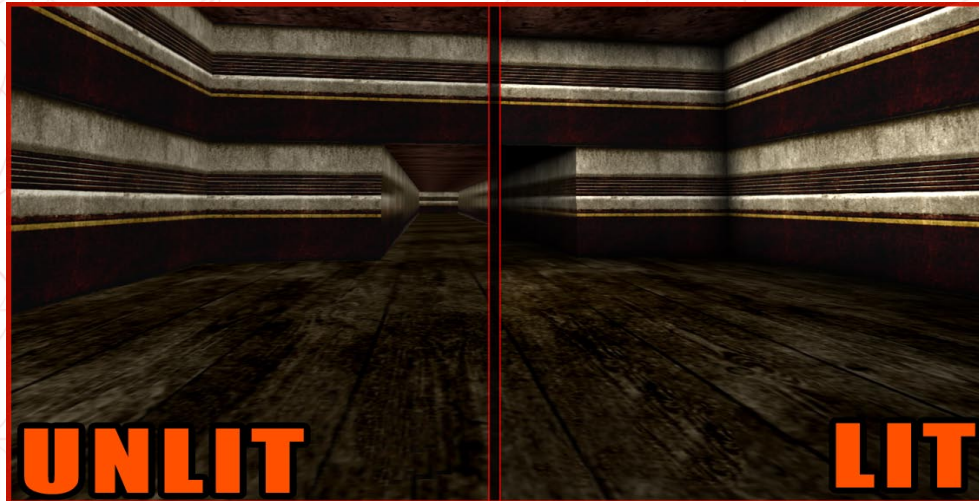


Image AV 75

One solution could be to raise or lower the intensity of the light but this is not recommended. In the end the fault lies in the texturing and not in the lighting so the textures are the feature that needs to be fixed. Fix the cause, not the result.

Changing the light intensity will also cause trouble if the level uses a combination of dark and bright textures, for example, a snow level with dark buildings. Lowering the light intensity would cause the darker textures to appear even darker. Raising the intensity of the bright textures would look way too bright. Therefore the textures used in an environment should be balanced and have roughly the same level of brightness!

The same is true for colors in textures. The colors used in textures will influence the look and feel of the lighting. It is essential to choose the light colors while texturing the level. If the textures in an area are, for example, very orange and yellow, it will look strange if they are later lit with blue lights, as seen below.

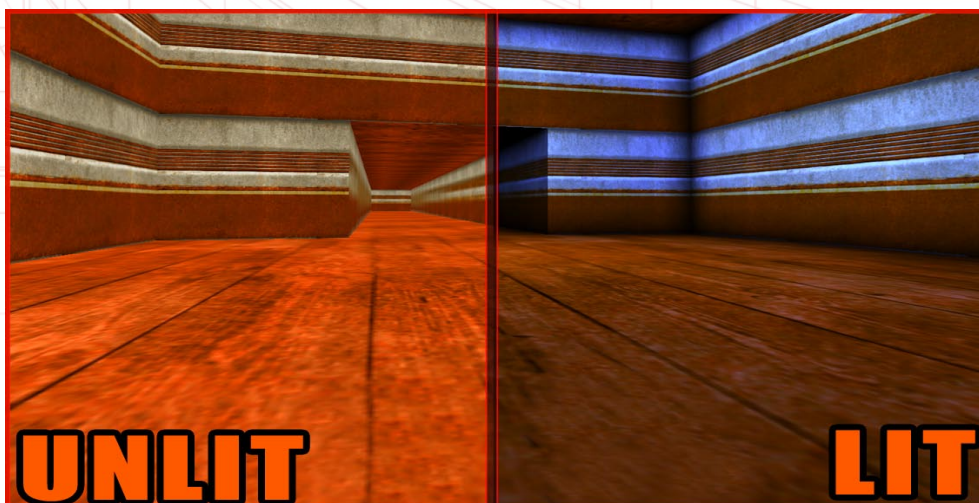


Image AV 76

If the design requires that the environment be lit with blue lights for whatever reason, then, during texturing, the textures choices should be those that are desaturated enough or have roughly the same color as most of the lighting.

The point is that the texture choice can heavily influence the lighting. Textures and materials are the base for lighting, and if the texturing isn't in harmony with the lighting, then one of the two is going to suffer. Remember that all the elements in a world are interconnected.

REALISM AND THE EXCUSE

Designers often light areas with very white light and without clear sourcing and excuse this by calling it realism. Such lighting not only looks bad but it is also the opposite of realism.

As explained before, white light is rather rare from a realistic point of view. Due to radiosity, dirt, age, and materials, light almost always has a color. Light is bounced around and from the moment it hits a wall or other surface it takes part of the color with it. The eye might not notice, but most lights are always slightly colored one way or another other.

Ambient light in games also isn't realistic. Too much ambient lighting can wash things out and make them appear flat and dull. There is a large visual difference between ambient light and proper radiosity lighting. Simply adding lots of ambient light won't make it look like radiosity lighting. Although the recent introduction of Ambient Occlusion mapping can help, one should still be careful about using too much ambient lighting; especially in engines that do not support Ambient Occlusion.

Radiosity lighting in the real world features very subtle differences in brightness. It is these subtle differences that ambient lighting cannot offer, and that ultimately make the result look artificial.

It is also more difficult to create a good looking environment with radiosity than one may think. This is because of two things: materials and sharpness. As explained before, a certain level of contrast in the environment is necessary. In real life this contrast is often partially due to materials. Environments with a lot of radiosity often gain a certain amount of contrast from subtle changes in the materials. For example, subtle reflections and a few shiny elements can make a kitchen environment appear interesting even though it might only receive indirect ambient lighting.

In games, however, this is still largely impossible. While the situation is definitely improving, materials and shaders often are still too simple or imprecise. There's a good chance the environment will end up looking dull if the materials are not advanced enough and if they can't be compensated for by using interesting lighting.

Until games support advanced and detailed materials and full Ambient Occlusion and Global Illumination, the designer will need to compensate for the lack thereof with more interesting lighting compositions, even if those might be a little unrealistic.

In addition to materials, sharpness can also play a significant role. While this, too, is a feature that has improved greatly over the past few years, and will continue to do so, games are often still not able to display very subtle details like small reflections and subtle variations in the lighting. Items like blurry televisions and monitors, low quality settings, and weak hardware, especially in terms of memory, aliasing, and so on significantly reduce visual detail. For the time being, games are far too imprecise to show subtle variations and thus the problem lies within too subtle materials or lighting variations may not be noticed or possible in a game environment. It is often best to exaggerate everything a little in games. This is also true for lighting. Until games manage to render photorealistic scenes, make sure the differences in the lighting is noticeable! Exaggerate it a little!

SOUND

INTRODUCTION

Sound is an often underestimated and forgotten element of level design. I personally have played dozens of games and levels where I could barely hear any environmental sound at all. While hearing is a person's second most important sense it is often ignored or given little attention in level design. Adding audio is not just simply adding a few obvious sounds to a level. It's about bringing the level to life by creating an interesting composition of sounds.

While silence is very rare and unrealistic, most games often still have more silent spots than areas with sound. In the real world pretty much no location is truly silent. One always hears something somewhere. Everything makes a sound, and a virtual environment should reflect this. Think about what would be heard when walking through a building if it were real. Wind surely wouldn't be the only sound, would it?

Sounds have two purposes. They aid and enhance the gameplay, as described in the gameplay portion of this book, and they are meant to enrich and aid the execution of the theme, story, and atmosphere. Without sound it would be much more difficult to create an ambience as the visuals would have to bear the full weight.

USAGE

Many levels have only one sound in an area. For example, a fan that makes a fan sound in room A, a ventilation grate that makes an engine sound in room B, a wind sound near a window in room C, and so on. While this kind of sound usage is better than using no sounds at all it still isn't exactly good.

The problem with the given example is that there are not enough sounds. The environment would end up sounding empty in many spots which is unrealistic and ruins the atmosphere, and thus the player's immersion in the environment. Another problem is that it would quickly become repetitive and thus boring due to the fact that the few available sounds don't mix enough. The result would be very flat and single-layered while the optimal effect would be one of depth and complexity. Place multiple sounds that play simultaneously. Overlap them! This can be done either in the engine itself or by mixing audio tracks in an external program. I personally prefer to mix sounds in the engine as it is easier to tweak them later on without going to the hassle of reimporting the files, although it can put additional strain on the hardware.

It is essential to know that there are two main types of level sounds, both of which should be used.

The first type is the true ambient sound. These are general sounds that have no direct source; they are indirect. They aren't very loud and they have large radii. The second type is the detail sound. They have small radii, higher volume and a direct source.

Logically, true ambient sounds are used to give areas a moody background sound such as some wind, distant birds, trees waving in the wind, distant war sounds, etc... They are intended to easily add sound to large areas and, because of their large radii, prevent silent areas. They are sounds that have no direct source.

Detail sounds, on the other hand, are used to accentuate details in the environment. For example, a machine in a room would have an engine sound, as would a truck waiting outside a building. Another example could be the sound of dripping water, a waterfall, a fire, a steam

vent, and so on. They are sounds that have a direct and easily identifiable visual source.

Both types should always be used in conjunction with each other. The point is to prevent that very flat feeling of single-layered sound. Mix them up. Not just with different sounds but also with different types. Even if an ambient sound uses twenty different samples, it still has the potential of becoming boring since all the sounds are the same type and volume and they can all be heard constantly. Detail sounds are only heard every once in a while and are louder which makes them less repetitive. Think of the composition tutorial. Without variation one ends up with a very flat line, different elements are needed in order to stand out occasionally.

There is a third but less important type: dynamic sounds. Dynamic sounds are those that are only played occasionally and, preferably, randomly. They are a mix between ambient and detail sounds. They add variation to the environment, yet most of the time they have no clear source. An example would be a howling wolf in the distance or a church bell that rings every now and then.

Dynamic sounds can also be triggered directly by the player. I often link a sound to some old wooden planks or rusty metal so when the player walks over it an appropriate sound will be played. This is great for the player's environmental interaction and it can add to the gameplay.

SOUND VARIATION AND PITCH

Make sure that there is enough variation within the sounds themselves. If the engine allows, experiment with forgotten sound properties like pitch. When I use the same sound multiple times in the same area I often give those sounds a slightly different pitch so none of them sound exactly the same. I put them out of sync with each other on purpose. If multiple entities/actors that cast the same sound are too close to each other they will only increase in volume due to their combined strength.

This should be avoided because the high volume that's created might become dominant and it will sound like the noises are coming from everywhere, which destroys the 3D effect. By using slight variations such as a modified pitch this can be prevented.



Image AV 77

In the example above, both sound actors play the same sound but with different pitches. The result is that the sound has much more depth. It is continuously mixed with a variation of itself.

A heavily modified pitch can also produce very nice results. By lowering the pitch of a sound significantly, it can sound completely different. I once created a scary ambient rumble in a level by slowing a happy African drum sample down to a very low pitch. Experiment with the sounds!

THE POWER AND STRENGTH

Nowadays PC's and other multimedia systems often have powerful sound hardware and software. Be sure to utilize this strength. 3D surround setups are common these days so make sure a level supports it. Source the detail sounds in obvious ways and make sure they're spread out evenly.

Many people's PCs and consoles also have the ability to support advanced sound features like EAX. If an engine supports it be sure to use it. Big rooms will need an echo, for example, and small wooden rooms sound different than medium sized metal ones because of the way the sound waves bounce based on the surface types.

Power and strength also refer to the volume of a level. Often sounds aren't played loud enough and therefore sound very weak. Take thunder and lightning as an example. In many games these barely sound stronger than some of the ambient sounds in the distance. Why? When lightning strikes nearby in real life the sound of the thunder is always much more powerful than other environmental sounds. The audiovisual goal of level design is to create an impressive and interesting world. Sounds that impress and blow the player away just as much as awesome graphics are practically a requirement! A dull sound simply won't do, so be sure to scale up powerful sounds, as far as can be done without making it annoying.

When something explodes the difference in volume compared to the other environment sounds should be clear and, obviously, louder. The same goes for many things: crashing aircraft or trucks, collapsing buildings, waterfalls, etc.

They are all loud and powerful so blow the listener away with them! Movies manage to do this, why can't games?

It can be a good idea to temporarily disable other sounds while the big sound occurs, simply to accentuate the big sound even more.

IMMERSION AND ATMOSPHERE

INTRODUCTION

Immersion and atmosphere are often underestimated and forgotten, but they are also powerful. The word “Environment” in level design means a world that lives and breaths through the use of design in all its aspects: architecture, sound, lighting, texturing, events, and so on. An environment can only be created by using multiple design tools. A few nicely skinned environment models cannot become a real environment because they don’t offer any immersion or atmosphere by themselves. Without lighting or sound to support their placement they will be nothing more than a group of three dimensional objects. Without immersion and atmosphere the environment is dead. It needs a soul and identity.

The ability to create an atmosphere and add certain feelings to creations is one of the aspects that sets humans apart from computers. A computer may be able to generate environments but without the help of people it won’t succeed in giving that environment a certain style and atmosphere. Immersion and atmosphere are also the only aspects of level design that can lift it from a type of design to an art form. It has the ability to convey emotions and feelings to the player in the same way movies can.

Good atmosphere and immersion makes an environment far more memorable. Experiences stick in one’s mind better when they are associated with certain emotions the player may have had in their own life. One may not remember the standard concrete corridors once played in long ago, but those atmospheric surroundings of the first *Unreal* or *Half-Life* levels, or that one really scary event with a monster, are memorable. It is these emotional connections that make players remember a level.

BASICS

Before an environment can breathe atmosphere and immersion, it must have a solid base. If the base is too weak or feels artificial it will be much more difficult to create an atmosphere. If players don’t believe in the situation they won’t be sucked into the story or experience the scary areas as scary. After all, why would a player be scared if a place is unbelievable and comes across as so fake there’s nothing be afraid of!

The key to immersion and atmosphere lies in presenting a level in a believable way. This does not mean the level has to be photorealistic but it does mean that the style, theme, and story are displayed in an acceptable package. Adding multiple different themes and styles close to each other, such as a demonic castle next to a Greek temple, is not usually acceptable. Thematically, they simply don’t work together and therefore the credibility of both will be diminished. Make sure the themes complement each other and blend smoothly. Avoid architecture with large empty areas or geometry that feels as if it’s about to collapse, lights that contain all the colors of the rainbow for no reason, monsters that just idle in one place, and so on. Avoid making the player doubt the logic of the world. It is for this reason that I am against the use of too much humor in a game, unless it’s one of the game’s key features, or other aspects like bad voice acting. Imagine walking through a really scary depressive area when suddenly there’s a funny poster or the character makes a bad joke after performing a certain action. This will completely destroy the credibility and atmosphere. Most people do not make jokes when in a very scary or dangerous area. The last thing a player wants when they’re feeling like they could die any minute is to hear their character making lame jokes. .

Prey was horrible in this respect. In *Prey* the player and his girlfriend, and for that matter half the planet, were abducted by aliens who use humans as food, and the entire game looks dark, cold, and scary. Yet the character still managed to make stupid comments on just about everything. What mentally sane person would make stupid jokes in such an environment and situation? Do not destroy the atmosphere that so much work went into with some stupid one-liners! If the situation is serious, but there's still a need to humor to the level, do so in a subtle way. For example, the occasional poster with a subtle message could be used, as long as it's not overtly obvious.

HISTORY AND SIZE

If the player is to be immersed in a world it must contain a certain amount of depth and unity. If all the levels are completely different in style or execution, and note that I did not say theme, it won't feel like one world and thus it will be harder to create immersion. Show that it truly is one world by ensuring the levels are coherent: share the same art, style, execution, and shared elements.

Creating mountains at the horizon will convey the feeling that there truly is a whole world out there. Returning elements such as landmarks can help remind the player that it's the same world, as explained in the Singleplayer Gameplay chapter.

This is also the case for history. If the feeling that the environment has a past and thus a history can be conveyed, this will prevent the player from feeling as if they are the center of the world and that the world is built especially for them. Evolution and time can also allow features to appear more logical and therefore less questionable. Having the player play through various situations and moments in the same area, as described in the Singleplayer Gameplay chapter, might be a good way to do that. After all, if the level appears to change over time, and as a result of other creature's actions, it increases the chance that it will seem like a living world.

MOVEMENT

This is also true for movement. Offering the player a seemingly living world by using animated decorations will greatly increase the believability of the level. Unless the theme or concept requires it, a static world will feel artificial, especially in regard to nature. Static trees aren't natural. Players will quickly notice this and start questioning the believability of the world. Create machines that really function, tree leaves that wave in the wind, clouds that drift, and so on.

The big difference between a rendered image or a photo and a game level is the motion and interactivity. Make sure the environment is more than a pretty render, let it offer more; add movement.

Movement is sometimes also an absolute requirement for atmosphere. Without falling rain it's impossible to realistically portray a depressing street area and without decorations like old creepy weathered boards slowly moving in the wind it would feel a whole lot less scary. Movement supports the atmosphere.

DETAILS AND CLICHÉS

Again, this also goes for details. The player needs to be offered certain details in the environment in order to support the theme and better experience immersion through credibility. For example, if a few murders took place in a house then it shouldn't look like a regular house. There would be details scattered around that help the player believe some terrible things have happened. Details such as patches of blood, knives, furniture showing signs of a fight, and so on. The player won't believe the murder story if they enter a regular building that looks very normal.

Besides credibility, the details also aid the atmosphere. If the same murder example is used then players should be scared and should feel the pain in the area. Therefore enough details should be present to experience this. The same type of details that help convey credibility also work for atmosphere and one would preferably also have other objects like photos lying around, candles burning, and any other element people usually associate with the related atmosphere and feeling. In this case it could mean a closed door with light shining out below it or a light bulb swinging around and/or flickering. If there's a knife with blood lying on the floor it is not only a required detail but it also scares people. A knife and blood is frightening and certainly so if the two are next to each other. Shards of glass might also help, as they can demonstrate that a fight took place, and shards of glass are sharp and dangerous, adding to the dangerous feeling.

A large part of thematic execution in level design is about clichés. The players do not have the time to think a lot. Most of the time they don't even want to and often the technology does not yet have the necessary power to try to explain a non-standard concept. Anything shown to the player must be simple and easy to recognize at all times. When they see object A they must immediately experience feeling B because A is commonly associated with B. It's up to the designer to identify and add such common associations to the level; in short, the clichés. What are the first and simplest things one thinks of when someone says scary? Happy? Cozy? What has been seen a dozen times before in movies or games in similar situations?

Clichés might not be original but they are used so much because they work so well. The reason why they are clichés in the first place is because of their quality. They are the best in their category and are the most efficient in doing what they do: conveying a certain feeling to the player.

The power of atmosphere is that much can be done with just a little. Just a single sound, color, or a simple moving object can work wonders depending on their use and implementation.

STORY AND DEPTH

Stories give depth and connectivity to the world. A good story sucks people into a world and keeps them in it; it adds emotion, depth, history, and fleshes out the characters.

In general anything that shows signs of complexity will add much needed depth to a world. Anything that shows there is more behind a house or a tree than just a mesh with a texture on it is good. Signs of age and events that previously occurred, background stories, evolution, reasons why it is at that spot, and so on are good. For example, a house might have been built on wooden pillars because the place used to be a swamp twenty years ago. A tree might have been burned down in the past because of event X which will act as a signaling event that X really did occur. This is also explained in the Singleplayer Gameplay chapter. If the environment has depth the player will find it more interesting and believable.

For the game *The Chronicles Of Spellborn* I first created an ocean level where the player experiences a sea battle and then, a few thousand years later, the player walks around on the bottom of that same sea. Through an apocalyptic event the sea dried up and all that remains is a humid rocky desert with lots of erosion, geysers, and large skeletons. While the environments are totally different, I did manage to connect the two and make this connection clear to the player. I added wrecks of the same ships the player played on in the old world, plants that resembled sea weed, as well as all kinds of other sea plants and skeletons, and sand textures that were inspired by a typical beach texture.



Image AV 78

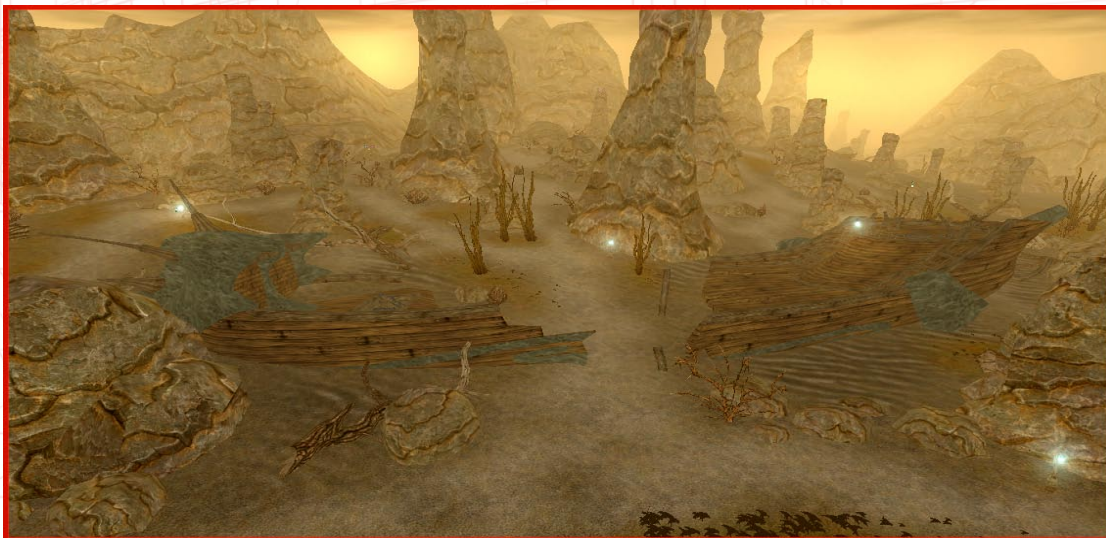


Image AV 79

It is this connection that unifies the world and gives areas their own history as well as explaining why the related objects are present. Essentially, the designer makes a big thing out of several small elements. And doing so will help immersion.

One note regarding stories: If there is to be a strong story then the core story should be communicated as efficiently and simply as possible. Far too many games present the player with endlessly long text to read through or spoken text/dialogue that lasts hours. The average gamer either doesn't want to wait that long, aren't readers, or simply aren't proficient enough in the game's English/other language to understand or enjoy it due to their age or nationality. Thus, the best bet is to explain the story through cinematics or events. They are

to the point, efficient, and far easier to understand. A picture is worth a thousand words in any language and for any age.

This does not mean, however, that the story must be simple and short. It means the basics of the story must be explained well. After which, one can add to the rest of the story as extra material. Those who are truly interested will read it while others who are less interested still can ignore the fluff, but still follow the core story.

The bottom line is that showing events is better than describing them in text or audio.

IMMERSION AND DEPTH IN MULTIPLAYER

Many people claim that design aspects like atmosphere, stories, history, events and so on are only good for Singleplayer levels or Massive Multiplayer games but this is not true. In the end, no matter what type of game is being played, it's always located somewhere – it has environments. And an environment simply needs these elements.

As long as it does not obstruct the gameplay there's no reason not to add them to multiplayer. They will improve the game experience. Multiplayer levels can have a story attached to them in order to explain where the player is or to provide some background info. Multiplayer levels can also feature signs of evolution or history and they can come across as either scary or welcoming. They can even contain events to support the story or theme.

Far too many times I've played multiplayer levels that were completely focused on just the gameplay. Even these levels are in need of good atmosphere and immersion. For one of my Capture The Flag levels I let a friend with a writing hobby write a very extensive background story to explain the history of the house the level takes place in and why certain things are the way they are. To support the story, or the other way around, I created a very spooky yet calm atmosphere in the house. The player knows that absolutely nothing will happen, as it is multiplayer after all, but, on the other hand, they do feel that something is going on, or happened, and if that interests them enough they will read the story. This, in turn should give them more understanding of the world they play in and thus improve the immersion.

In another multiplayer level I created I had a couple of old World War II Spitfires flying around the building that contained the playable area. The planes were attacking each other around the building and they regularly shattered some of the windows of the rooms. The sequence added a great deal of extra depth and a much increased wow factor. The impression made on the player was increased. While the planes were in fact completely irrelevant to the gameplay it did enhance the complete game experience and it helped the level stand out from the mass and be memorable. And that is good!

AUDIOVISUALS

THE CONCLUSION

For everything explained in this book there are always exceptions and situations where element A or color B should or should not be used. This book is not intended to be a definitive set of rules to confine design to. Instead, it should be treated as more of a guideline that explains the basic concepts and principles behind all the elements involved in creating virtual environments. It aims to provide a deeper understanding into why certain decisions are made; the right perspective and frame of mind.

In the end, no amount of knowledge can replace actual experience. A good amount of information will surely help but without plenty of practice and ample patience one cannot master the art of level design. It takes years before most level designers get to a point where they don't do things because they know it will work well but they do it because they know why it works well. Being able to create a cool-looking and great-playing level is only one part of being a level designer. Understanding what it is exactly that makes it look cool and play well is the second. If one can understand why X will result in Y, then there will be that much more control over the direction of the project and the end result, which is good. The outcome will be more direct, cohesive, and consistent with a smaller chance of failure.

Don't start work on a level if the end product isn't fully realized. Nor should work be started if the available time, skills, and assets have not been thoroughly analyzed. Know exactly how, why, and what before starting any work. Design is a function of the available time, skills, and assets and is not blindly following a dream, idea, or concept. It might be too much, or even impossible, to create.

Strive to critically examine the concept and the length of a project and evaluate it. Never forget that above "good" there is always "great". Push the effort that extra step up to the next level of quality. This is not only true for the initial concept but also for the finished design. Work on a design shouldn't end simply because it looks good. Instead, analyze it and improve it until it's not just good, but is great!

One of the most important things to remember is that level design consists of a wide range of elements such as lighting, composition, and architecture. All these elements need to be balanced and consistent. They must be made for each other. Element A must support element B and element B must exist to support element A. Without one, the other cannot exist. All the elements must feel, and look, as if they belong to the same universe. Each one must be designed with the entire world in mind at all times.

Harmony and balance are two very important words in level design. Different aspects must be in harmony at all times. At the same time, however, they also require adequate variation. And variation needs to be precisely balanced! Balance element A and element B in such a way that they provide enough variation but also ensure that neither gains the upper hand, nor damages other aspects of the game or mod such as the framerate or the overall goal. Level design is about balancing a wide range of elements until the environment is in perfect harmony. Provide enough gameplay differences to please both new players and veterans, demonstrate enough visual difference to make the environment shine, run fast, and yet remain consistent with the game's overall setting. It is all about compromises.

The gameplay in a level should complement the core gameplay of the game or mod. Encourage the level to take the entire game to a higher level of quality. Let it enrich the way people play the game or the mod and let it provide extra depth. Give people a reason to chose that level to play the game in. Make the map gameplay desirable but never drop the support of the core gameplay. Don't try to steer the game in a whole new direction with just a level; accept the core gameplay as it is delivered.

Never forget that levels should be fun and interesting to play. Avoid making the player feel as if they are not in control unless the story absolutely requires so, for example in a prison situation. Make the player feel as if they can control the outcome of the story and the game world and that they can choose their own direction, even if they can't. And be sure to always keep it fair toward the players. Don't allow the players to die without also letting them see the danger ahead of time. Also, don't intentionally let them get lost. In general, avoid creating any situation that can result in irritation and frustration because it will make the players stop playing.

Make the levels impressive and memorable. There are tens of thousands of levels on this planet. Make each one created stand out from the others in some way. Do something special; something that makes an impression on the players and for some reason sticks in their minds.

Finally, remember that static and dead worlds are a thing of the past. A modern virtual world is an environment that should reflect an atmosphere, a feeling, a style, and a history. It needs to tell a story; not through text, but through the gameplay and visuals. A level is a living world with dynamism and emotion.

I hope you enjoyed reading through my book and may it help you make great levels in the future.

If you are interested in working in this industry, be sure to give my other book, *The Hows and Whys of the Games Industry*, a look!

APPENDIX A - LEVEL EXAMPLES

In this section there are three levels that I made: Sae, Redkin, and Krodan. All the levels are deathmatch levels for either *Unreal Tournament 2004* or *Unreal Tournament 3*. There are pictures of the levels in various stages as well as an extensive explanation detailing what I did and why I made certain choices regarding the design or the visuals.

I work best with strong organization. This means that I work in a certain order in terms of all the different elements that a level is made of. For example, lighting comes after texturing and I will never add lights before there are enough textures around. One aspect always forms the base for the other. If there aren't enough of the previous elements, then the required base starting point is missing.

All of these levels can be downloaded for free from my portfolio website, and can be opened up in the *UT2004* or *UT3* Editors for those who wish to examine them in detail.

SAE

An *Unreal Tournament 2004* level

DM-Sae was built for *Unreal Tournament 2004*'s MakeSomethingUnreal contest. The goal was to show just how much one could do with a minimum amount of time and assets. The level was one of the least time-consuming levels I have made, taking me just over seven part-time days from start to finish.

THE BEGINNING

I didn't start the level with the idea that I wanted to make a cavern. I began with the fact that I wanted to make a level with a minimum amount of time available. I had to find the most efficient theme possible.

After some brainstorming I ended up with a cavern because it requires a minimum amount of art assets, it is relatively quick to model, yet it offers great atmospheric potential. As explained in The Checklist I then needed to refine the theme and gameplay. There are dozens of cave themes out there and I had to pick one. A tropical cavern was out of the question because I had already made a tropical cavern map in the past. An ice cavern would have been good but a friend of mine was working on a similar theme and I didn't want to compete with him. In the end I concluded that I had to choose between a lava cave or a water cave. I eventually picked the water theme because water has more atmospheric potential and is more original than lava. *Unreal Tournament 2003*, for which I originally created the level, already had several lava environments but nothing really wet.

That decided, I now needed to get a clear picture of what a water cave looks like and, especially, how mine should look. I found a few dozen images on a picture search engine and got a clear idea of how a cavern looks pretty quickly. However, I also had a few large questions left over after looking at all the pictures. One of issues was where the light would come from. I couldn't go for a realistic setting that's very dark with a lot of ambient light because it wouldn't look good so I needed to find something else. I looked for something that fit the theme and looked fairly logical yet provided enough highlights to make the environment shine. I eventually went for orange emergency flares that were dropped by divers and plants that give off yellow light along with a strong blue ambiance.

Another issue was that my walls would look very flat and boring if I were to make a realistic cavern so I again had to find an element that could break up the walls yet fit the theme. I initially thought of adding prehistoric paintings to bring variation to the walls but I eventually went for a special type of fern. Ferns are not a hundred percent realistic but they did fit the situation best. Ferns require a wet and dark environment and my cavern offered that. I went for plants because paintings are too flat and thus don't have the required effect. They would not have succeeded in filling up the rooms up as they are too flat. So although the paintings were the most logical choice I didn't use them and I went for a pretty illogical choice of plants instead. Logic is not always the best for your visuals. The trick is to make illogical features appear logical by blending them into the environment.

After I further refined the visuals I came to the conclusion that the level was lacking special elements: landmarks. Without them, the environment would have been too homogeneous. This is why I asked a friend to model a rock statue of a woman for me. The statue gives the area more identity and helps the players navigate. Another idea I had was to add large waterfall to the level but due to the level's small size I felt that this wasn't the best choice. The waterfall wouldn't have the space it required to feel big.

The gameplay was inspired by the theme. The intention was to encourage fairly standard gameplay because there was little time available thus I avoided experimental gameplay. I also wanted a few areas that were influenced by the water in order to connect the theme more with the gameplay. I'll cover this in more detail in the gameplay section of the example.

The level was going to stand out because of the amount of polish I planned to put into it, because it was a unique and fresh environment, and because of its atmosphere.

Coming up with all of this, plus the floorplan, took me around two hours.

THE FLOORPLAN AND GAMEPLAY

The floorplan was mainly designed in my head but I also drew a quick sketch on paper. The floorplan had to be simple and small because of the restricted amount of time. It basically ended up as a circle with one big room in the middle and two smaller rooms on the sides.

I didn't make any type of test layout or whatever else for this level. I started modeling it from the moment I was happy with what I had in mind. My experience allowed me to do this, and there was not enough time to take any extra steps.

As mentioned before the level had to have water that influenced the gameplay. I did this by adding plenty of puddles and shallow water areas, and by placing the armor powerup underwater so that players would have to take a dangerous dive into the water. The armor dive also offered another escape route out of the main room, albeit a slow and dangerous one.

The level only had one deep water area because otherwise the gameplay would be slowed too much.

I also added a second floor in one area in order to break up the height a little because having everything on the same level is boring.

GEOMETRY

I modeled the entire cavern as one mesh in my 3D package of choice, and vertexpainted the floor to allow the sand to blend into the rocks by using the vertex colors. I have some tutorials on these subjects on my website.

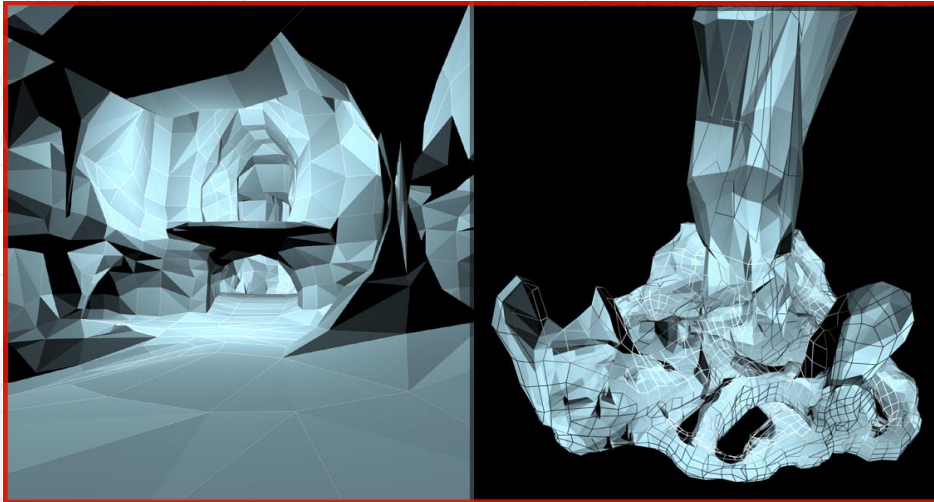


Image EX 1

I always try to add an overwhelming and bombastic feeling to my levels but the problem is that with multiplayer levels, and certainly ones of this size, it is not always easy to capture that feeling; especially when there's only a week to work with. That is one of the reasons why I had to implement the statue: to try to make the area look more impressive. The statue and waterfall behind it point up. The vertical movement helps the impressive feeling.

I raised some ceilings and added holes in it for the same reason, and also to break up the flat ceilings. Variation is important. Some people suggested placing ancient ruins around to add more special elements but I didn't have the time for that.

Once I had the cave mesh, I imported it into UnrealEd and applied some very basic lighting.

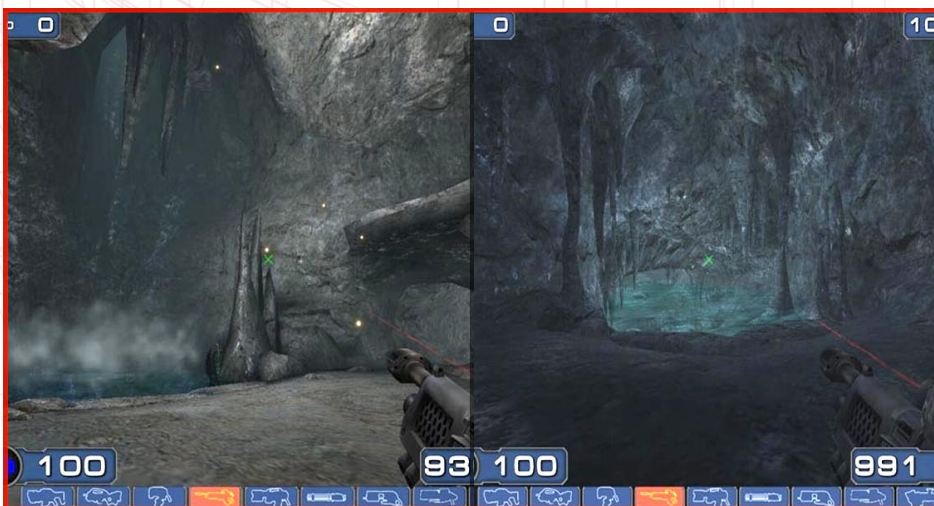


Image EX 2

Aside from the main cave mesh, I only used a very limited amount of art assets. I basically only had a couple of low-poly stalactites, two plant meshes, a flare mesh, the statue,

and related content. Time didn't allow for a great number of assets to be made and the environment didn't need it either. The entire level with all its content was only 7 MB.

On the contrary to what some people believe many, or complex, assets aren't necessarily needed for an environment. How something is used is often more important than what is used.

TEXTURING

I used vertex colors to blend the cave textures and to create smooth transitions.

The textures used in the level are all relatively simplistic. The rock is just a tiling texture with a shader on top to add some properties like specular to attempt to simulate a wet look.

The textures all have a blue overlay on top to improve the consistency and to help them fit the wet theme better. Water is commonly associated with the color blue. Therefore adding a blue tint to the textures connects them more solidly to the situation. If I want to use a couple of textures that have widely different colors I simply adjust the colors slightly to pull them together.

LIGHTING

For this level I used turquoise and some orange/yellow. I also used some whitish-blue occasionally. I used the turquoise variant of blue because the level is cold and wet. Blue is a cold color and commonly associated with water. Turquoise is even better at this and therefore the color would reflect the theme.

The yellow and orange colors were chosen because they form a nice contrast with the blue, as explained in the lighting chapter. I used two tones of yellow/orange because I also had two types of light sources, plants, and flares. It allowed me to add in a little variation.

While I usually never use truly white lighting, the reason I did use some sort of white for this level was because I needed a third color for yet another light source. In addition to my standard plants and flares I required another special color for the two large spotlights near the statue. I added these lights for two reasons. One, it would have been repetitive to only use turquoise and yellow lighting, although I was careful to only use the third color rarely in order to prevent the level from turning into a random mess of light colors. The second reason was that my statue had to stand out more. It was shrouded in darkness so I needed something to brighten it and grab the player's attention.

These two problems convinced me to add the spotlights. I decided to use a very light white-blue light since white is a neutral color and therefore wouldn't conflict with the other colors. It also feels somewhat cold and it is a realistic color for a spotlight.

After I applied some basic lighting the cave quickly evolved in something better looking.

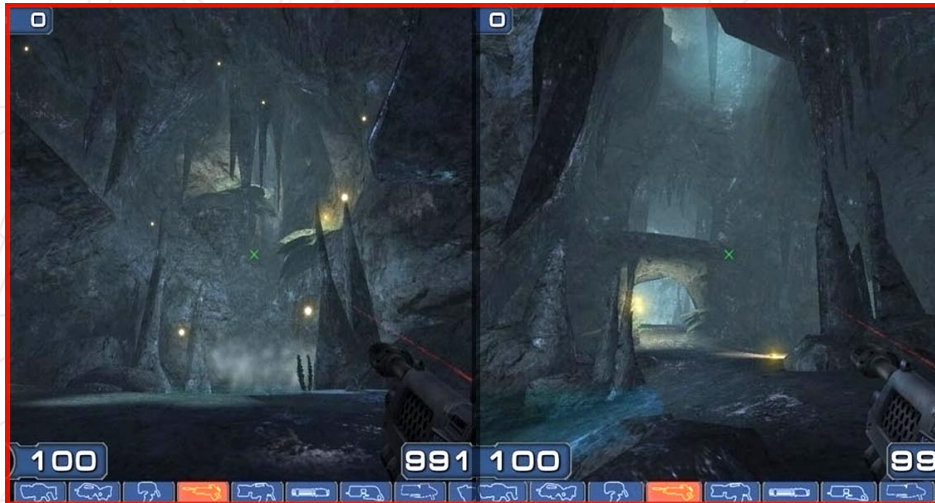


Image EX 3

I evenly spread the light sources throughout the rooms in order to create nice compositions. I did not make large areas with the same color, brightness or darkness.

THEME, SOUND, ATMOSPHERE, STYLE AND SPECIAL FX

I quickly got an idea of how caves should look when I searched for reference pictures of caves for this level.

I made the choice not to add such things as ropes hanging down, scuba gear laying around, sea creatures, and so on. Even though I was in need of detail, I wanted to minimize the human influence in this environment. Human influence can make things look cheap if not enough time is put into it. I didn't want to take that risk due to my time and skill limits.

While the plants aren't very realistic, they look better than other possible solutions. The primary goal is always to make something look good, not necessarily completely realistic; even when making a realistic game. Realism is often too weak to impress the viewer; it often requires extra elements to lift it to the next level of quality.

I used many particle emitters in this level to simulate a wet and foggy environment, which is something with which the light-blue distance fog also helped a great deal. The level has several types of particles such as foggy spots, clouds of moisture, dripping water, fireflies, flares, and so on. I needed all of them in order to show the player that the environment was alive and not a static picture. Motion! I needed the seemingly illogical fireflies because some areas were too empty and I needed their yellow color to create a contrast to balance out the large amount of soft blue with some strong yellow.

FINAL THOUGHTS

Before applying the finishing touches to the level, I quit working on it for a couple of days and gave myself some time away from it in order to give myself a fresh perspective. My “beta” levels usually are already very polished and clean, so by the time I reach “beta” I only have very minor issues left.

The level was made in only one week’s time and only during the evenings. I succeeded fairly well in my objectives. It won a few prizes and nominations in the contest.

The final result.



Image EX 4

REDKIN

An *Unreal Tournament 2004* level.

DM-Redkin was meant as an indirect successor to my *Unreal Tournament 2004* level DM-Rankin.

THE BEGINNING

I created this level simply because I felt like making another industrial Deathmatch level. It was not intended for any project or contest.

I originally aimed to make a low poly, but very fun, standard gameplay level, along the same lines of what Rankin offers. However, throughout development it evolved into a more serious and complex level full of features and special elements. A basic level wouldn't have impressed anyone and I realized this after about a week's work. This was a design error and I should have realized and foreseen that rather obvious issue before I started.

One of the features that I wanted to implement was some areas that overwhelm the player; areas that have the ability to impress the player. Adding such rooms can be quite a challenge in this type of level. Multiplayer levels often tend to be quite small and impressing the player usually requires a large amount of space.

In the end, I decided to add a large, narrow canyon outside and one high room inside. The canyon enabled me to expand the world well beyond the borders of the level and to give the player a spectacular view of a canyon. The canyon is narrow to make it appear deeper and more vertically aligned, and thus more impressive. The shape and alignment of geometry really does make a difference.

The next design issue was to figure out how I could make a canyon look interesting. I had several ideas but all of them had issues. One of my ideas was to add a large battleship in the canyon but then I realized that the canyon wasn't wide enough to support such a large ship. Making the canyon wider wasn't an option because I needed the vertical alignment for the previously mentioned reasons. The next idea was to add a submarine bunker at the end of the canyon with a sub coming out every few minutes. That idea, however, had the same problem. Even a submarine was too big for the canyon.

In the end I decided to add a small factory and a waterfall at the end of the canyon. Waterfalls are always impressive to look at and they are also very vertical so it would reinforce the vertical feeling of the canyon.

Once I had agreed with myself on making a large canyon with a waterfall I faced a new issue. A canyon with a waterfall was not enough. The level needed something extra, something that moves and impresses. The battleship I originally wanted to add would have been great for that but, as explained, it wasn't a good option.

Another idea I had was to add huge elevators on the sides of the canyon that transport crates and other materials but that didn't fit the canyon very well either and with looping all the elevators I could run into technical challenges.

Since the elevator idea clearly wasn't going to work very well either I moved on to another idea: having the Unreal Warfare dropships fly through the canyon and land at a landing pad.

The pad however would have been too big to fit in the canyon so that wasn't a possibility either. To fix this issue I scrapped the landing pad from my design and decided I could just let the dropships fly through the canyon without landing. Later on, this evolved to letting spitfires fly through instead of the futuristic dropships. Spitfires seemed more original to me,

more stylish, and it also allowed them to be armed.

In addition to all this, I also decided the level needed a lot of moving machinery. Moving elements are very underestimated and underused in most environments so I wanted to bring about a change in this.

The level didn't have a time limit. There was no asset limitation and apart from the spitfires there were no technical challenges or aspects I was inexperienced with. The design was solid and allowed for both good gameplay as well as good graphics. I had a pretty good idea about what I wanted to make, how I wanted to make it, and how it would look at the end.

There was nothing that could block the level from being made. The design was efficient and possible.

THE FLOORPLAN AND GAMEPLAY

I made up the floorplan in about an hour and a half mainly by simply brainstorming it. I also made a really quick paper sketch to set up the basic layout. After I had the floorplan set I immediately started to build it in the game's editor. It took me around four hours to make a low poly yet playable representation of the floorplan.

Contrary to what some people expect I didn't playtest the floorplan at this time, but immediately started working on art assets. I've made so many *Unreal Tournament* levels that I don't need to go in-game anymore to find out how it plays. I'm already so used to the scale and the speed of the game, that I can estimate how it will play quite well.

I chose to make a medium-sized level because I wanted a level that would play well for both 1-on-1 and Team DeathMatch since Rankin offered the same. Too large a level also takes up too much time.

The floorplan generally flows well. It has no dead end areas, except one intended one, and each area has multiple exits. It has two or more floors almost everywhere to add a good amount of Z fighting and several shortcuts, as well as a shield jump area.



Image EX-5

I tried to implement a large number of features to give the gameplay as much depth as possible. I often work with sounds and moving elements to warn players of nearby enemies.

The panels of the screenshot open wider when someone uses the jumppad. They act as an extra warning for other players since they draw attention with movement and sound. They also improve the interactive feel of the level.

I also made use of standard elements like wooden planks and water puddles in this level. When players walk over them they will give away their position because of the sounds.



Image EX 6

I also added windows into some inner walls. It allows the players to look into the next area which can help to give away the enemy's position.

The level also had to resemble its big brother Rankin which is why I included the small basement, traversable air vents, a dead end corridor with a powerup, a long stretched-out corridor with one double floor area, and an elevator.

The idea for the flowerpot shortcuts, the connection points between two floors with wooden planks on flowerpots, was added later on when I noticed the possibility.

I also added a few small traps at the end. They were mostly used to improve the interactivity of the level and increase the number of small details around. The traps are very small and do too little damage to seriously influence gameplay. If I overdid them it might have alienated more serious players. In one place, for example, a player can open a small vent which will blow non-lethal spray of hot steam at another player.

GEOMETRY

The geometry is composed of more than a thousand mostly generic and modular meshes and hundreds of BSP brushes.

The interior's basic skeleton was made completely out of BSP and looked like this after I had just started.

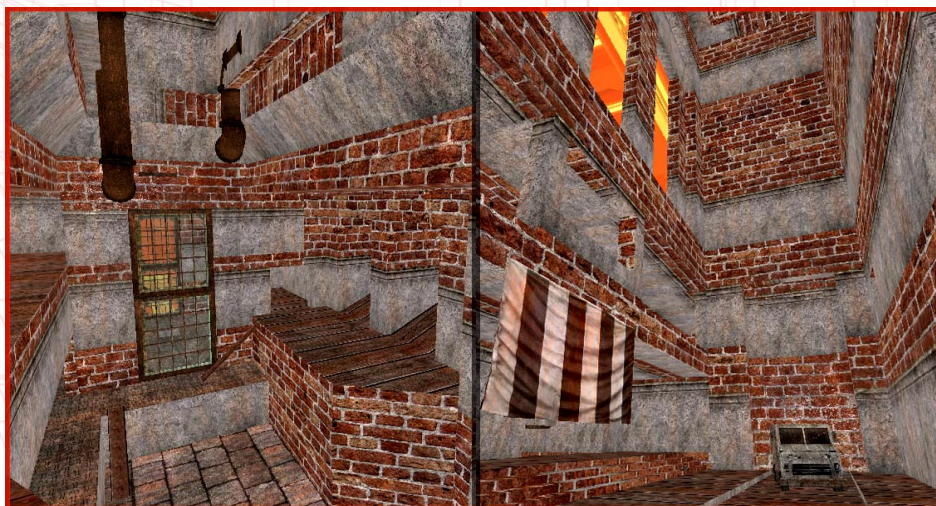


Image EX 7

The entire exterior was made with only meshes. All the meshes were made independently in a 3D package and almost all of them are generic or modular. They all align to the grid and they are all easy to reuse in a wide variety of situations. Apart from a few rare exceptions I almost never use unique meshes that only fit a single area or situation. The walls in this level, for example, aren't huge unique meshes but instead are many small modular blocks. I can make my walls as long as I want without having to model extra pieces and without losing a

lot of time, by just repeating the modular meshes.

The only non-generic and modular meshes in the level are the canyon, the building and pipes seen in the distance, the forest, and the two large pipes in one of the rooms. All the other meshes are either generic or modular.

Think creatively and beyond the initial purpose of a mesh. I used a random generic pipe mesh from a retail *Unreal Tournament 2004* package as a rain pipe and trimmed its edge with an ancient floor tile from a different package. This kind of mesh use can also be seen by opening up a *Gears of War* or *Unreal Tournament 3* level. Some seemingly unfitting meshes are used in such a way that they blend into a wide range of situations.

All the meshes I used are low poly and use simple textures. While the models are very simple it still looks cool. What is done with something is more important than what it is! In a manner of speaking, one is better off spending more time on how the meshes are used than on modeling and skinning them. If the time is available it is, of course, always better to do both; polishing both the meshes and the level itself to the maximum. But as that's not always possible, aspects like good composition and lighting can have a big impact on the level. They can compensate for simple models. Brilliant art assets are not absolutely necessary as long as their lower quality is compensated for by other features.

I initially wanted to go for architecture with more curves and smoother transitions but that proved fairly hard to do with the modular workflow.

TEXTURING

I went for a red and white theme composed of mostly rusty steel and bricks. I also added a couple of warning-stripe textures as they are quite stylish. They add extra definition and style to the place.

I made sure the level looks good even when unlit. The textures are the base of the lighting. If they look like a mess without lighting, the situation won't improve after the lighting is applied. I always have a clearly defined color palette before I start and I stick to it to prevent myself from running into trouble later on.



Image EX 8

The textures were mostly self-made and most of them were photo-sourced. Again, all of the textures have a color overlay on top. In this case, it was an orange overlay.

The World War I-style posters were added to make the place look more stylized and surrealistic. Inspiration for this came from other levels. Empty brick walls can look boring; objects like the posters can break them up a little.

For this same reason I also added a few logos around. Note how the logos are subtle and fit the environment. They actually enhance the environment rather than disturb it. I have seen many clan and company logos in levels throughout the last few years that didn't fit the

environment at all which is something that should be avoided at all cost. Consistency is holy.

One of the design tenets I tried to show was that with only a few new textures and elements one can turn something old or overused into a new environment. There are many levels in *UT2004* that are very similar to Rankin simply because they all used the same textures. Using just a few new textures can make something look refreshing almost immediately!

LIGHTING

The lighting in the level is primarily composed of a strong orange combined with a colder turquoise. I used a strong orange color instead of regular orange/yellow because I was aiming for a very warm and dusty feeling. Red was not an option since it is too aggressive. The turquoise color had to act as a contrast for the strong orange and cool things down.

My third color varied from green to red. The lake in the canyon and the computer panels cast a green light while several machines and the van's rear lights have red lighting. I gave the machines red lighting because it such a powerful color. The machines are one of the features of this level so I had to make sure people would notice them.

Although strongly limited by the vertex lighting of the engine I also tried to establish a decent shadow composition, mainly through the use of wooden planks, windows, and grates. The lighting can create nice shadows if it hits these types of objects.

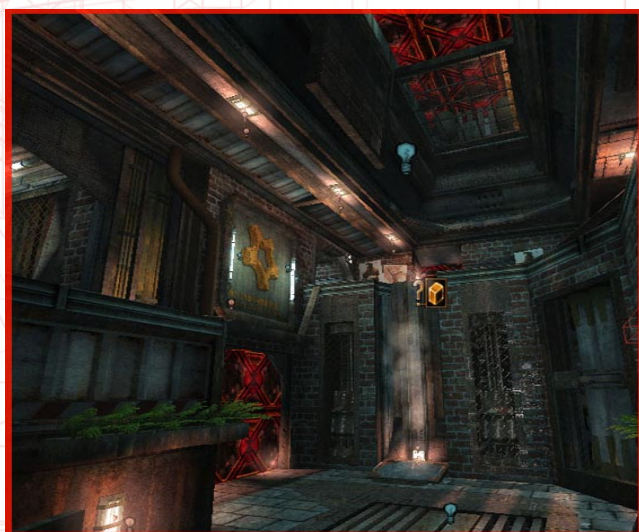


Image EX 9

The level quickly reached a more finished stage after basic lighting was applied.

At this point, if the level is examined closely one will notice that the shadow direction doesn't match the position of the sun. I did this because the default sunlight direction didn't influence the interiors much at all. I wanted the sunlight to cast into the interior areas. While it is not logical, it does look better if the shadows fall in the current direction. As mentioned before: it is more important to make something look good and logical at first sight than to make something

100 percent realistic. I don't think any player actually noticed that the shadow direction does not match the sun's position yet it does look better like this.

Now one may wonder why I didn't just put the sun on the other side so that it does match the shadow directions. The problem was that I couldn't do this and I predicted why before I lost time trying it out. If I were to flip the position of the sun, then the big pipes on the canyon walls would have received bright orange light, just like the factory in the distance. This wouldn't have been as attractive because the lights on the pipes need to be in the shadow in order to look nice. If the lights received sunlight most of the contrast would have been lost which is something to avoid. Flipping the entire canyon and all of its content also wasn't an option as that would be too much trouble.

For the canyon lighting itself I added two sunlight actors. One as the real sun and the other one pointed in the opposite direction with a second color and less brightness to function as a type of directional ambient lighting.

THEME, SOUND, AND SPECIAL FX

One of the aspects that had to set the level apart was the machinery. I added about a dozen active machines, such as several large wheels, and two groups of steam pumps that supply jump pads with steam. The level also has four huge pumps in the vertical room.

I added the machinery to set the level apart from others; to add more movement and dynamic features to the environment in order to impress players. A large pump is impressive to see and it livens up the area. It transforms a static picture into a living world.

Far too many of the levels out there are nothing more than static pictures while, in fact, they should be alive. Certainly in the future this will be the only feature, aside from interaction, that will set game environments apart from renders and paintings. The strength of game environments is not walking through pretty-looking static images: it is playing in a living world.

The level contains many industrial ambient sounds; some of which have a very low pitch. There are around a hundred sound actors/entities in Redkin even though the level is only three rooms large.

The spitfire planes outside were meant to give the level a special touch. They had to enhance the style and atmosphere, give the environment an extra dynamic element, and, above all, help the level be more impressive. Any element that can help a creation to stand out from the masses is good.

I initially tried to make the planes with a regular mover actor but I quickly encountered all kinds of limitations. For one, the limited amount of eight keys/positions was not enough and the movement ended up too jerky because movers don't support smooth transitions between keys. I ended up choosing the most radical method: a mesh animated in an external 3D package. I'm not an animator at all so it gave me quite a lot of trouble and the movement still is a bit jerky but at least it ended up much better than if I had stuck with movers.

I let my friend Desperado skin the planes and I attached smoke and bullet particles to them. In the end I made a few mistakes with the net support code but, all in all, it works fairly well.

I also added some extra small details when the level was nearly finished: cables hanging down, papers scattered over the ground, water hoses on the ground, and bottles. Small details can greatly enhance the look and style and they don't require much time to place. They enhance the theme. It should be clear that the environment is alive and that activity is taking, or has taken, place.

Also, objects should be connected. A simple example of this is the water hoses. At the end of the hoses I added a water puddle. The two elements become one; each one justifies the existence of the other. This is also the reason why I made the planes fire at the building that contains the playable area. If the building appears to take damage, it will not only look cool when the windows shatter, it will also connect the planes with the playing area by showing they are not just decorative objects in the background. They are part of the same world and each can actually influence the other. This is the kind of connection a designer should be always be reaching for.

FINAL THOUGHTS

Only a few serious bugs popped up during beta. Some particles refused to work correctly, and the spitfires had synchronization trouble online. These were also the only problems I got stuck on during development.

In the end the level took me seven months of part-time development. The final result looks like the following:

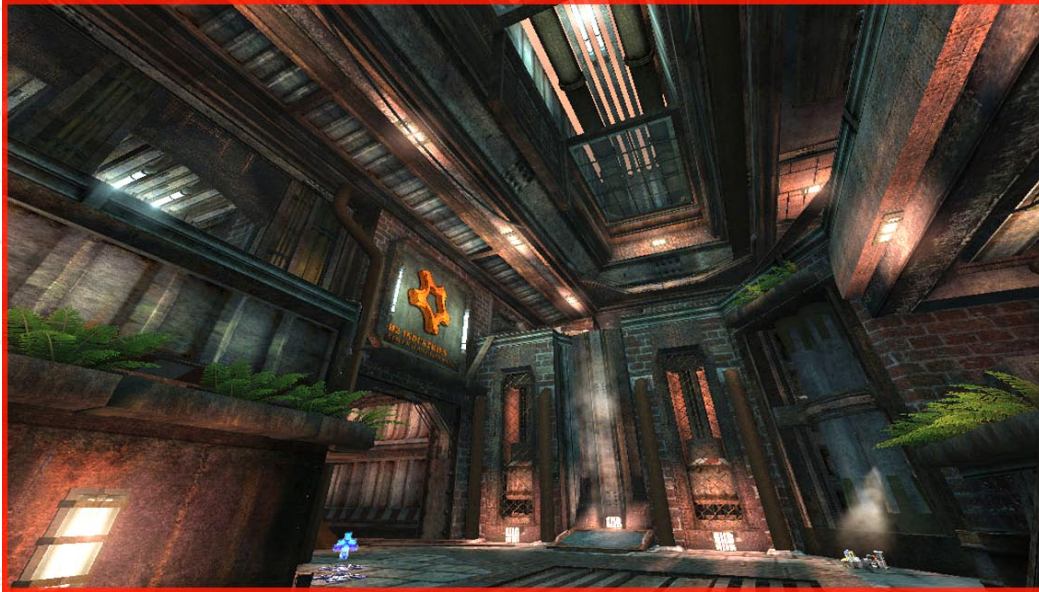


Image EX 10



Image EX 11

KRODAN

An *Unreal Tournament 3* level.

DM-Krodan is a small but highly atmospheric outdoor DeathMatch map. It took me around a month to create all the models and textures, part time, but just three days to do all the editor work including all lighting, materials, particles, placement, optimization, and so on.

THE BEGINNING

The level was originally made for a *UT2004* project that, unfortunately, died. It was also originally meant to have taken place during a thunderstorm, with a helicopter crash site, and a number of dead “survivors” scattered throughout the level in and around some tents

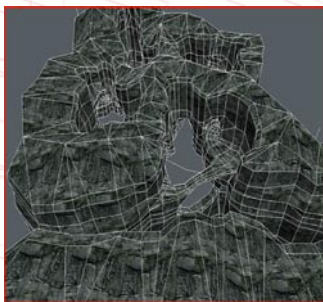


Image EX 12

waiting for a rescue that never took place. When I finally started working on the level again, after having it left untouched for over a year, I decided to redesign the whole theme, and add in a massive landmark: the house. I decided that the level was really missing a big and important element to give it an identity, and that a bunch of tents with bodies around weren’t going to be enough, so I decided on an Art-Nouveau inspired house. I also deepened the canyon and raised the surrounding rocks to make the whole level more vertically aligned. Vertical alignment dwarfs the player, increasing the dramatic feel of a place.

THE FLOORPLAN AND GAMEPLAY

The floorplan was relatively easy and, just like Sae, it basically resembles a circle with four areas placed in a circular shape. Its gameplay is fairly simple, and the focus is on casual gameplay. The level consists of two floors and an area with a deep pool of water. There is a wooden platform in the water with a powerful pickup on it. Players need to jump down into the water to grab the pickup and then climb back out of it through a single exit making it quite a dangerous endeavor for them to undertake. Not only does the water make them an easy target, the single exit will make anyone’s movement in the area very predictable. Getting the item comes with a heavy price.

The house follows the same idea. Even though players can’t access the house, they can walk up to its front door where they find yet another very powerful item. The area is a dead end though, so players are forced to leave the house’s porch as quick as they can because they become trapped as soon as an enemy shows up.

The floorplan was designed on the fly, which wasn’t terribly difficult considering the simple nature of it.

GEOMETRY

The entire level was modeled in a 3D package, with no brushes being used at all. The rocks were modeled relatively low-poly, with a well-tiling normal-mapped rock texture contributing all the detail. As already explained in the Composition section, there is a lot of hidden logic behind the geometry with the majority of it being vertical lines. The house itself is also very vertical, and the glass porch is used to break up the otherwise too flat building. The glass porch sticks out from the house, so that if it were observed from the side, it wouldn't look like one big flat surface. For the same reason the window on the first floor is recessed into the building, and it has a little balcony which further breaks up the otherwise too large flat wall.

The large round window on top was inspired by an Art-Nouveau house in the city of Antwerp, which had a very similar large round window. The round window is supposed to grab the players attention and force it upwards and to give the house a more refined look. If

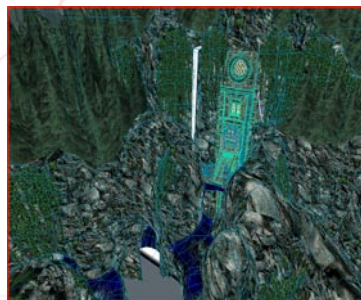


Image EX 13

I had added a regular window instead, the house would have looked too generic and standard. I wanted to give it a specific look so it would become more recognizable.

On both sides of the house there are two 45-degree walls connecting the house to the rock wall behind it. I added these 45-degree walls to make the house appear more connected to the rock wall behind it. Simply sticking the house into the rock with a straight wall might have made it look too generic, as if it was just a mesh pushed into the rock.



Image EX 14

TEXTURING

Most textures in the scene were composed of photos I took myself, photos I found online, and photos I found in free texture libraries. All of the textures have a dirty green overlay on top in order to reinforce a consistent look. The green color originated from the original concept; a dark and very wet forest environment. I decided to stick to this color palette even after I restyled the area.



Image EX 15

I decided to add strange flowers to give the area a surreal feeling. One wouldn't expect flowers, commonly associated with happy thoughts and summer environments, in such an environment. The clichés discussed earlier don't always have to be added and, in certain circumstances, the opposite can be done in order to reinforce a certain feeling. This is more difficult though.

The rock materials received special rim lighting, causing their outer edges to glow, thus accentuating their silhouettes. Although not entirely realistic, it fits the style, and helps add

a sharp feel to the level and synchronizes the feel of the materials with that of other textures; for example, the pointy, sharp-looking grass texture, and the pointy pine trees.

LIGHTING

I used a rather standard orange and blue light scheme because I wanted to make the level look fairly warm, thus making it appear more welcoming, for which the orange lighting was essential. I had no need for experimental or original light colors as the general feel and composition of the level were strong enough to support the level on its own. There was no real need to try to make the level stand out further with more original and much more difficult color combinations.

The level uses four different sunlight actors, all pointing in opposite directions, to soften the shadows slightly and acquire a more realistic light setting. If I had used only one or two primary light sources, the level would have had the typical, artificial-looking strong shadows. Perhaps more importantly, it might have unbalanced the visuals in the level because the geometry is quite high and blocks a fair amount of light. If the light was cast from only one side it may have created too much shadow.



Image EX 16



Image EX 17

Even the shape of the lamps fit the rather curly Art-Nouveau style of the architecture. A lamp is not just a lamp; it can be more than just a square. Large dark areas always have a lamp somewhere in order to break up the darkness.

In one area there was barely any sunlight hitting the rocks at all, and I was stuck with a very large dark wall. I ended up adding a gate with a car behind it - a car from the 70's to make it look even stranger and more surreal - and gave it strong, white headlights to shine into the area. This immediately broke up the large dark wall and it broke up the otherwise very similar

lighting in the level. The bright blue headlights are the third light color, just like in Sae.

THEME, SOUNDS, AND SPECIAL FX

The theme itself is rather unrealistic and non-specific. It is the presentation and the composition, however, that makes it look believable and interesting. Again, what is done with something is more important than what it is. Would anyone ever build such a house there? Unlikely, but for the visual value it has, it works great.

Aside from the visual potential of the theme, I also chose this environment because of efficiency. It is relatively easy to create natural environments as assets can be reused more often, and it all has to be placed slightly less precisely. Creating a "next-gen" environment alone is quite a lot of work, as explained in the Design chapter. It is important to pick an efficient and attainable concept or it will never see the light of day. Krodan is actually quite simple: it only has one big unique mesh, the house, only about twenty relatively simple meshes, the big rock mesh, and it only uses a handful of textures. It has little content, yet it works. That's the way to go.

The level has dozens of sound actors, to ensure that there is no truly silent spot in the level. The crickets in the grass also have varying pitch levels. Aside from these points, the sound work is rather standard.

There are also quite a few particles and different types of fog. First of all, I added birds flying in the air, which are basically just animated textures that project onto a particle sprite. The birds add some life to the level.

The most important particles of the level are probably the large plumes of smoke that rise up from the river deep below the level, illuminating the crevice below the house. The crevice was too dark and uninteresting so, instead of placing lights in it, I added a particle that also brightens the area. The particles themselves also rise, adding to the vertical movement of the overall composition.

FINAL THOUGHTS

I am very satisfied with the way the level turned out in the end. It even ran better than the official levels because of the minimal amount of assets I used.



Image EX 18



Image EX 19

APPENDIX B - INTERVIEWS

This chapter features interviews with several established designers and artists from the games industry. Each interviewee talks about a subject of their choice. For example, one talks about texturing while another focuses on architecture. The aim of the interviews is to gain insight into experienced peoples' philosophies and their preferred way of working.

Note that the opinions of the interviewees may not reflect their employer's stance on an issue. Their answers must be viewed as their personal opinions.

CEDRIC FIORENTINO – GENERAL LEVEL DESIGN

Cedric Fiorentino started as a modeler back in 1994 and quickly evolved to a Level Designer while working for Epic Games where he worked on all the *Unreal* games. Known in the industry as "Inoxx", he's known for his involvement in the community, especially for the release of the "Inoxx Pack", a free level pack for *Unreal Tournament*. His most famous level is "Facing Worlds", the most played *Unreal Tournament* level.

In the past he also worked for Webzen as the Environment Director on the MMOFPS *Huxley*, and he is currently working on an unannounced project for Arkane Studios.

How do you begin and plan a new level? How do you work and in what order?

This depends greatly on the particulars of your company, your team, the style of game you're working on, the technology you're using...

The actual process of making levels evolved quite a bit over the years. It used to be a task that could easily be done by a single Level Designer, just opening up the editor, start cutting blocks and brushes, add some textures, drop a few lights and you had a Level. Some of the levels that shipped with *UT* were in near-final form after just a day of work.

The process became gradually more and more complicated as the games increased in complexity. The time required for a single Level Designer to actually build all the elements in a level quickly became prohibitive, so Level Designers started using libraries of already built assets. This method became a standard about five years ago.

Then it became obvious that the level designer tasks were still too time-consuming, so next-gen engines further subdivided the tasks by allowing multiple Level Designers to work simultaneously on the same world. Nowadays you can easily have one person working on terrain, another on modeling assets, someone else working on sound and one working on the gameplay. And so on.

These days each world is basically a collection of layers, or sub-levels, that can all work together seamlessly.

I'll explain the workflow that I've implemented for the project I'm currently working on.

The creation of a new level nowadays involves quite a bit of team work. During the initial stage we decide the basics and choose the goals we're trying to achieve with this specific level.

First of all, we need to choose the gametype, the overall size and the number of players required to play, as well as the maximum number of players. Once these factors have been decided, we start building a basic shell. The shell is a gameplay prototype with little or

no visual detail, build out of simple geometry, to represent an accurate design of the final gameplay. It is important to us to have the game playable as soon as possible, in a prototype form. Once the shell is playable we playtest it daily and redefine it multiple times in a short period of time, with daily play sessions and daily improvements.

In the mean time we're deciding what's the environment is going to be like, how it fits in the rest of the world. The concept artists are sketching some roughs, they get done when the shell gameplay has been refined.

That's when preproduction stops and actual level production starts. The team is divided in Cells, each cell is made of a Level Designer, a Texture Artist and a Modeler. The Cells get the Shell and the sketches and start detailing the Level while keeping the gameplay consistent.

The last stage consists of performance and memory optimizations as well as visual polish.

Each level has its specificities, depending on the skills and inclinations of different team members, or external factors, but we try to avoid diverting too much from this ideal process.

Once all that is complete, preproduction stops and the actual level production starts. We divide our team into cells. Each cell is made up of a Level Designer, a Texture Artist and a Modeler. Each cell gets a prototype level shell and the designs and sketches and starts detailing the level while keeping the gameplay in mind.

The last stage consists of performance and memory optimizations as well as visual polish.

Each level has its specificities, depending on the skills and inclinations of different team members, or external factors, but we try to avoid diverting too much from this process.

What do you try to achieve with your levels and what do you do to try to attain that? What's your philosophy on levels, your goal?

There's only one thing that's really important when you are working on a game: Fun. In the end that's only thing that matters. You can have the most beautiful environment, but no one will play it if it's not fun.

Of course other than fun there are also a lot of other things to achieve in a level such as re-playability, visuals, immersion, unique environments and so. There also a lot of technical challenges, for example, trying to make as much as possible fit in the limited memory budget, get the best possible framerate and offer a lot of scalability, so that people with slower computers can still have a great experience playing the game.

How do you develop a certain style or art direction for a level? How do you decide what style and atmosphere a level should have?

The overall look and feel is decided by the Art Director, the Lead Concept Artist and myself. We just sit down together and discuss the specific details of a level. What will make it unique, how it will fit into the world, where it's located and what the historical and environmental factors of the storyline are that might affect the look of the level. Other elements can be considered as well. For example, depending on the uniqueness of the level, we need to decide which assets have to be created, which assets can be reused and which ones can be outsourced.

This is a very different process from the "old days" where a single Level Designer could single-handedly make a level from start to end. The reality of today's game development is such that all these measures are necessary in order to get levels done within an acceptable timeframe.

What do you do to maintain the consistency between different elements and the rest of the environment? How do you preserve the unity of the world?

Each environment is carefully sketched out by the concept art team, so consistency is easily attained.

The world is divided into several regions which have their specific look and feel. If a level is supposed to take place in one of these regions, its look will be determined by the background story of that world region and the specific properties of the level's location.

Consistency is also improved by the use of modular elements that can be reused within the same region, sometimes with minor changes or with no changes at all.

How do you view the connection between gameplay and the visuals/environment as a whole? What should support what? What kind of influences do the style, theme, and/or atmosphere of the environment have on the gameplay? Or the other way around?

Without a doubt, visuals are here to support the gameplay. Using the development process I detailed earlier, we can ensure that there is minimum friction between the gameplay direction and the art direction because those two are implemented at different times. Of course there are always small points of argument about what plays better versus what looks better but these contentions are fairly limited, as the intended gameplay layout is established early on.

What's more of a problem is the connection between the gameplay side and the technical side. It's more common to have a good, fun layout that requires changes in order to run well on the target machine.

Which aspects are most important to you when creating a level? What part of the process or element do you devote the most time to?

I tend to be a maniac when I make levels and I tend to spend way too much time on little details. I've learned how to get better at it over the years but it's still tempting to iterate on an area until it's perfect. I'm also very picky about performance, probably because I can't forget the frustration of playing games on slow computers when I was younger, so I really try to get the best performance, sometimes going through great lengths to save a few ms from a frame.

This kind of attention to detail is a very good thing for console games because any performance you gain will be enjoyed by everyone, any memory you save can be used to have more details somewhere else. On the PC, it's harder to justify spending so much time on optimization, because after some time, whatever you gain on one graphics card might be slower on a different model. And the hardware evolves so quickly that all the compromises you make at a specific point in time may seem silly just six months after the game's release.

In your opinion, what are some of the most common mistakes people make when creating a level?

There are many mistakes that are easy to avoid. One thing that really annoys me in some user made levels are 45-degree ramps. Even though they are easy to walk on in-game it doesn't look very realistic. Buildings in a real world never have slopes steeper than a few degrees. I think nothing looks more fake than these levels with 45-degrees slopes everywhere, it completely breaks the illusion for me.

Also, just because the computer let's you get away with all kinds of designs that would never support themselves in the real world, it doesn't mean that you can ignore the laws of physics. I've seen a lot of structures that don't make sense, pillars that look too small to support what's on top, or just above a hole, architectural details from different eras used together... and so on.

All these mistakes can be OK if done intentionally in order to create a stylized feel, but when you see those things in a level that is intended to look realistic; it breaks the illusion and credibility.

What do you think people should always keep in mind when designing a level?

Make it fun! Easily the best advice I can think of. Don't get stuck on a particular area, sometimes you invest so much time and effort into one area that you might become reluctant to change it later on while sometimes it might be in dire need of a change.

Another thing to keep in mind is what other people's experiences are going to be when they play your level. You have to think about the different ways to play the level so it appeals to a broad audience, while maintaining a good balance. It's very useful to just let others play your level, sit behind them, and watch how they play. Some people will find clever ways to abuse the level to their advantage, these problems will need to be fixes as quick as possible, or on the other hand some tricks or strategies that seem obvious to you might be too complicated or hard to find for somebody else.

ROGELIO OLGUIN – GENERAL LEVEL DESIGN

Rogelio Olguin has made level designs for *Tactical Ops: Assault on Terror* and *Unreal Tournament 2004* for which he mainly worked on AS-Mothership, AS-Fallencity, DM-1on1-Desolation, and DM-Hyperblast2. He has been making custom levels since he acquired *Doom 2* and continued on through games such as *Duke Nukem 3D*, *Quake*, *Quake 2*, *Half-life*, *Unreal*, *Unreal Tournament*, and *Unreal Tournament 2003*. Rogelio has led several user mapping projects for the Unreal community. He makes sure every level he designs is unique and new to his previous designs with strong focus on artistic design.

How do you begin?

The first thing I do when I'm about to start a level is to figure out what kind of gameplay elements have to be implemented and also what the story and theme will be. Those have to be very clear from the start of the project. The concept stage is very important to me when I start a new level. Making sure everything is prepared before starting any models is always good.

Of course the concept will not be the final result since concepts also tend to not work in a level design sometimes. But they will offer a good base to start from.

Where do you find your inspiration?

I get my inspiration from everywhere. Really you should not trap yourself with one area. Inspiration comes from training your eye to see details in the world or in your minds eye. Though I do have a certain biased towards Art Deco, Victorian, and Art Nouveau.

How do you work and in what order?

The order I work with is pretty standard. I go from Concept to laying down the main forms and paths for the player than I start adding the actions in the levels such as elevators, doors, and other events that need to be implemented. Once those are set up I start on making the visuals.

First I start with textures making sure I use a good color palette for the level and what the mood will need to be. From that I build the filler details, like trims, lights, chairs, and anything that might be needed to convey the theme and mood of the level. Lighting is done throughout the filler stage. The end is the polish stage, making sure everything works.

A very important part with level design is to regularly have the map tested by others to see what they think. This should be done from the very start when you have your very basic blocked out layout of your level.

Why do you make certain decisions? What is your philosophy and what do you do to try to achieve that?

I make decision based on my traditional art teachings. I base everything on balancing the moving canvas, what is in view. I pay attention that every scene is well balanced visually and interesting to view although some decisions are made for you since game art is still restricted to the speed of computers.

My philosophy is form equals function. It is the philosophy of Frank Lloyd Wright and I whole heartily agree with it. The level layout must drive the player through it; not impair the player.

What do you do to maintain the consistency between different elements and the rest of the environment? How do you preserve the unity of the world?

This starts from the concept stage. Though, I tend to put more than one style on some levels to help the mood. City maps specifically have very different styles, building to building. It is good to maintain standards in your level designs but it is equally as good to know when to break away from the standards and make a design that is true to the naturalism of the real world. The real world buildings are refurbished all the time and sometimes in a single building you see traces of many different styles. Color/lighting is the greatest weapon to equalize the world. Color has been described also as the number one design element since color can directly influence mood and feelings. Color is a great symbolic tool to use.

How do you develop a certain style or art/game direction for the elements in the world?

This all depends on the concept stage. But concepts should always be upped a few times over when you make your level designs based on concepts. For example, if the lighting requires to be made more dramatic to improve the atmosphere then do so. A game design crew should develop a style together and find what works and what does not and apply it. It is a collective direction. Concept can only go so far but the real challenge is upping the designs and to keep improving them all the time.

Which aspects of design are most important to you? What part of the process or aspect do you devote the most time to?

I devote a good bit of time to the concept stage; it makes level design far easier if you have a good idea you can work from. Apart from the concept stage I also devote a lot of time to the visuals. I want my levels to have an essence of symbolism within the colors, textures, lighting, and forms. I want people to unconsciously experience them and the atmosphere and style they bring forth.

What's the most commonly made mistake?

Not having a good concept before starting and working on one area at a time. Finishing area per area, one at a time, will usually make the level look scattered and inconsistent.

What should people always keep in mind?

Level design is based on nature. Everything in the level should have a feeling like nature has had a say in the building or construction. Nothing is left untouched by the human hand or nature. For me being a level designer is to embrace romanticism.

ANDREW WELDON – SINGLEPLAYER GAMEPLAY

Andrew Weldon originally hails from the *Half-Life* mod community, most notably as a contributor of two official levels to *Natural Selection*. He joined the *Quake 4* team at Raven Software in 2004 and served as a level designer for the singleplayer part of the game and also contributed to a number of multiplayer maps released in the post-release map pack. He is currently working for Gearbox Software on the game *Borderlands*.

How do you work and in what order?

First and foremost, there has to be a plan. What's the goal? Is the level going to be an epic singleplayer adventure? A scary horror story? A sprawling coop level? I usually start with nothing more than that, scope and game type. From there, I might sketch up some rough layout ideas or dive into prototyping. I like to keep any sketches pretty simple. Just rough boxes with some arrows here and there can do the trick. For me, at least, overly detailed layout sketches hinder the sort of iterative evolution that comes through the entire design process. It's like working from a checklist, it'll help identify what needs to be done, but won't really give it the love it needs.

Instead of committing to a weeks-long project only to find half the great ideas don't work, prototyping is a great way to test out and tweak ideas in generic settings in just a few minutes or hours. And it's a lot of fun! I fiddle all the time with little prototype test maps to see how different ideas will work out. This phase can help weed out bad ideas, emphasize really good ones, and also give important information on workload. Getting deep into a project only to find that an element of it is going to be completely unreasonable can be devastating. Identifying issues like this early opens the opportunity to trim things down or at the very least plan accordingly.

When it's time to start building, rough geometry is best. It's good to get in and get a feel for the size and check out how it feels to move around. No sense wasting time with detail when you don't know if you'll scrap an hour, or even a week, of it later. Ideally, all gameplay would be fully implemented before building a single detail, though plenty of tweaks will occur all the way to the end.

From there and into the visual phase, I actually get pretty free-form. Where some (and perhaps more intelligent) designers work in distinct phases of texturing, detailing, lighting, etc., I tend to start by completely finishing an area or two to establish the theme, then bounce around the level as ideas arrive in my head, eventually propagating out to the entire map. It's a little dangerous in that it leaves a little more room for error, but I like the sort of visual evolution that comes out of it even if it means a little extra work to go back and touch up other areas to keep the consistency.

What are some of the aspects you focus on in your levels?

Single player requires a certain level of linearity to tell a story, but that doesn't have to mean a straight path through the level. It's fun to offer multiple paths or even just set up some rooms to be entered multiple ways with the AI inside reacting differently for each. Setting things up this way adds a hefty workload, but has the payoff of giving players choices and adding at least slight replay value.

I'm also a big fan of reusing areas. This gets into the often hated issue of backtracking, typically seen running back and forth through the same chunk of geometry with monsters magically respawning in it each time. This can be done well, but I prefer multiple unique

passes rather than repeated ones. There are a lot of ways to push flow through key areas. It might be as simple as running under a bridge, then crossing it later. Perhaps multiple catwalks and ledges serve as connecting points in moving through a large vertical space and its surrounding areas. As a designer, smart reuse of areas yields a lightened work load and allows more time and effort to be spent on much more focused content. As a player, finally reaching a level-wide landmark can sometimes be reward enough for navigating the level while passes near and through memorable landmarks can help ease navigational confusion present in some more sprawling areas.

Where do you find your inspiration?

I get the most mileage out of the games I play. There are always those elements that can stick with you. For me, a couple include the intro tram ride in *Half-Life* and the apprehension of venturing into *System Shock 2* that can help you re-think the way you approach your work. Even the most mundane detail can kick-start your thought process into a new direction you might not have thought of before. Community maps are great in that regard, too. As time goes on, it's easy to get used to doing things a certain way. Seeing what others are doing, particularly in the up-and-coming talent, can break you out of that stagnation and into continued improvement as time goes on.

Games aren't the only thing that can inspire a level, though. In the past I've pulled ideas from books, TV, photos, and movies. Even walking by a random building can give that spark. There are also incredible resources online. Concept art galleries are wonderful inspiration (always be mindful of copyrights before taking too much inspiration). Stock photo galleries and urban exploration photos can also provide some of the most jaw-dropping real-world scenes imaginable.

What do you think people should always keep in mind when designing gameplay?

At the most basic level, playing a video game is nothing more than pressing buttons on a keyboard or controller. Gameplay at its simplest is nothing more than making the act of pressing those buttons fun. Much of this is handled in the core game mechanics but the important elements of pacing, context, and means are delivered solely within the levels. This yields some important questions: What is the player doing? Why is the player doing it? How is the player doing it?

The "what" covers the player's actions and gameplay pacing, the spread of these different actions through the game. Is the player shooting? Exploring? Sneaking? Driving? Jumping? How much time is spent with each action? Perhaps 10 minutes of all-out gunplay is balanced by a few minutes of explorative cooldown. A more in-depth puzzle could be followed by a cooldown of some brainless shooting. Good pacing balances the different play styles within the level (and even on the broader scope of the entire game) to keep any one aspect of the game from becoming too stale. This is especially important when trying to create longer stretches built around one element, say, a vehicle level. Being locked in to one thing for 30-45 minutes is bad pacing. The Halo series has addressed this well by not only breaking up extended vehicle sections but also employing a more open implementation of vehicles where players can sometimes use different vehicles or take on different roles within a vehicle (drive for a stretch, hop in the turret for a bit, etc.).

Context, the "why," is primarily story-side and can exist independently of gameplay. However, context can trigger extra emotions which can sort of "fool" a player into thinking something is much more than it is. A simple jumping puzzle can be boring and annoying. A simple jumping puzzle against the backdrop of crumbling geometry while being chased by powerful enemies is exhilarating! It's good to think of presenting the gameplay in ways that will garner that sort of emotional involvement.

“How” refers to the means by which the player accomplishes his or her goals. There’s some overlap with the actions relevant to pacing, but this time it’s in the context of what result comes from that action. An enemy can be killed by gunfire, or a puzzle can be solved by moving a physical object. Simple enough.

One consideration to keep in mind is today’s common aim of “cinematic gameplay”. It’s easy to take a “show the player” approach. Enter room X, then trigger a cinematic of enemies entering. Defeat boss Y, then cut to a cinematic of his death. While the presentation of cinematics and scripted sequences makes for an important element when present, keep in mind the players’ payoff in gameplay comes from partaking directly in the act. Robbing the player of his precious button-pressing time too often or in the wrong places can drastically reduce a level’s appeal.

How do you see the connection between gameplay and the visuals/environment as a whole?

It’s interesting; it’s possible for the two to exist independently, to have fun gameplay with no visuals or environment to speak of, or stunning visuals with no gameplay to speak of. Yet, particularly in an action game, the visuals can become an extension of the gameplay. Well-placed darkness and shadows can naturally lead into slower and stealthier play. The clutter, railings, pillars, the omnipresent crate (or whatever more creative solution may be applied), and other mid-sized details can become cover for more tactical play types.

On a less tangible level, a game’s visuals can even enhance that ever-important ‘feel’ of the gameplay. The whimsical and at times plain silly gameplay provided within Psychonauts was mirrored perfectly in the wild, stylized, and varied environments. Half-Life 2’s grittier and oppressive nature was accented wonderfully by the often muted and at times beautifully drab urban environment. Would either game have “felt” right if the environments had been switched on each other? In the end, while visuals are worthless without that elusive fun factor, visuals on top of fun with the right balance of presentation and style can make all the difference between a great level and a truly incredible one.

TOM HANRAHAN - ARCHITECTURE

Tom Hanrahan is currently a Level Designer at Monolith Productions. Before arriving at Monolith, he worked at Timegate Studios on the games *F.E.A.R Extraction Point*, *F.E.A.R Perseus Mandate*, and *F.E.A.R Files*. Tom broke into the industry based on the work he did creating custom levels for *Quake*, *Half-Life*, *Half-Life 2*, *Far Cry*, *Doom 3*, and *Unreal Tournament 2004*. He has been designing levels since he was 14 years old – the days when his mom wondered why he was getting a C in Geometry despite the fact he spent his free time at home dragging vertices across a grid.

How do you start? What do you always look at or design first?

If I'm making a level for fun and I don't have any idea what I want to do yet, I'll often start by opening the editor and designing a small test room or area. It helps me decide upon an architectural and lighting theme for the level. Once I'm happy with my theme, I'll come up with a layout for the entire level.

However if the theme is already in place and the concept artist has already dictated what I should be designing, (as is the case for most commercial projects) then I'll usually start by drawing a complete layout for the level. After that, I just need to make sure that what I create matches the scale and style of the concept art, and from there it's all just grinding it out.

Where do you find inspiration for architecture?

Everywhere. The best thing you can do as a Level Designer is to go to places! You don't necessarily need to visit the Guggenheim or the Parthenon; you just need to go out and observe how and why the structures all around us are built; look carefully at details and ratios and apply what you see to what you build.

I grew up in a big city and lived close to an even bigger city, so from a very early age I feel I was inspired by modern urban planning. Images of skyscrapers, bridges and subway tunnels were etched into my brain during my formative years. Of course, the places that you experience don't even need to be real. I am equally inspired by the architecture I see in computer and console games and the places I read about in books.

I find a lot of existing video game architecture to be very helpful, especially FPS levels. If I'm feeling uninspired all I have to do is load up some of my favorite levels that the Quake, Unreal, etc. communities have put forth and marvel at them for a while. Sometimes real life architecture just doesn't spark my imagination in the same way that a lot of FPS architecture does, which is probably because FPS architecture isn't bound by things like feasible engineering.

My own imagination is also one of the biggest sources of inspiration. Imagination is given a lot of fuel by personal experience so you have to make sure you're always keeping your imagination strong by continuing to seek out everything life has to offer. I'm always amazed at how inspired I'll become just by going out for a walk.

What's the most common mistake you see in level architecture?

The most common mistake I see today in amateur levels is a lack of detail. It can be very difficult and time consuming to propagate a level with the amount of detail necessary to satisfy gamers' expectations. It's tough because this is often the least "fun" aspect of level design – you've built already built the level, chosen a theme, and addressed gameplay concerns, now it's time to place detail items carefully around the world and add architectural flourishes like bevels, broken sections, and other assorted details that make the level feel more interesting. Some amateur designers simply don't have the time or patience to do that sort of thing and consequently release levels that have a distinct lack of polish. The best advice I can give on the topic is this: no matter how detailed you think your level is you can always add more.

What should people always keep in mind when creating architecture for their levels?

Make sure that the architecture you've created in your level facilitates fun gameplay! I can't stress that enough. It's nice that you have the ability to create an exact replica of your local supermarket but if it's not fun to put the player in that situation then no one will play it for more than a few minutes. Some of the best levels I've ever played make no sense from a traditional architectural standpoint. Always design levels such that the architecture fits the gameplay and not the other way around.

The other thing you should always keep in mind is that you will usually need to include some kind of "stereotypical" element to the architecture in order to sell it to the player without telling them directly what it's supposed to be. For example, if you're making a sewer environment, you'll want to make lots of pipes and pools of sludge everywhere. In real life, a sewer might not always look like that, but if your level looks like it has no function outside of existing solely for gameplay purposes, you're going to bore the player. You should always try to find a balance between the abstract and the exact – never make a carbon copy of a real life structure or area, but never make a level so abstract that the player doesn't know what the function of the area is.

How do you see architecture fit into levels in general? According to you, what is the role of architecture within an environment?

I believe that architecture has two main roles within a game level: creating a sense of place, mood, and purpose and then facilitating that sense of purpose.

Architecture is very important to defining the way a level feels as a whole. Even in 2D games where the architecture is nothing more than some platforms in the foreground and a static piece of art in the background, the way that those elements are placed, drawn, and set into their final position is vitally important to shaping what the players feel when they are playing the level. The style of architecture is extremely important for defining the context of the game, especially in first person shooters. If you look at *F.E.A.R.* for example, the architecture is fairly simplistic and true to what you'd see in any commerce-based area -- an office building, a warehouse, etc. In fact, I think that's why *F.E.A.R.* was so successful – the well-executed real-world architecture was vital to facilitating the challenging artificial intelligence that you encounter. I don't think *F.E.A.R.* would have had the same impact in a fantasy environment, or even a real world environment that had less of an "everyday" feel.

What do you do to keep the architecture consistent?

I like to have as much reference material as possible when I'm working to help ensure the architecture in my level is consistent. It would be really embarrassing to put something in one of my levels that was disparate to the style I was trying to execute. For example if I were tasked with making a historically accurate Romanesque level, I would never cap a building off with a minaret. Of course, I might try to blend those styles if I had no "accuracy" constraints holding me back, but as a general rule of thumb, reference material is an excellent guideline when it comes to keeping a consistent style.

If I'm left to my own devices to create a style (as I was for my *Half-Life 2* deathmatch map DM_Ortho) I certainly do like to try and mix a number of different architectural elements into my level.

It can be very tricky to tie them all together, but the best way to do so is to keep other elements of the level mostly uniform. For example, keeping the type of light fixtures and lighting consistent can help to blend contrasting styles. You also don't want to call too much attention to a single element. Subtlety is hugely important in keeping things unique but still consistent.

DANIEL LUKA - TEXTURING

Daniel Luka is a Senior Texture Artist at Eidos in Montreal. He started his game development career in his early teens when he first released custom created content for the game *Monaco Grand Prix Racing Simulation 2* back in 1999. After creating game art in his spare time for five years, using several different game engines he later moved on to become a renowned artist in the *Half-Life* mod community.

The breakthrough came when Daniel shipped his first game *Day of Defeat* at age 17 in May 2003, which had started out as a mod project. A year later he was picked up by Ubisoft Entertainment and has since contributed to established titles such as *Tom Clancy's Rainbow Six Vegas* and the *Splinter Cell* franchise.

In April 2006, Daniel was invited to be a guest speaker at the Savannah College of Art and Design where he spoke to an audience of 100 students and professors about what it takes to "become a video game artist".

After spending a year in sunny Barcelona, Spain Daniel is now back in Canada where he is working at the Montreal-based Eidos studio that is developing *Deus Ex 3*.

How do you begin?

Textures are a vital part of any computer game. They bring life and color to the world and if cleverly used in combination with lighting can evoke many different emotions in a player. When working on creating a realistic environment, for example, a real-life location my first step is to collect references of that location. This can be done either in the form of photos or videos. Additionally, in professional game development, companies often send their employees to these locations so that they can experience them first-hand and collect all the reference material that they will need during the production process. The artists then take those references and use them as base for creating their game art assets in accordance with the project's art direction and style.

Where do you find your inspiration?

I find my inspiration in many things. It can happen in every-day situations such as walking down to the grocery store or just going for a walk. I may see a nice little piece of architecture on the way and say "oh that's cool, I really like this!". Luckily we are living in a world where technology is getting more and more advanced and cameras are getting smaller and easier to carry around. I recommend to always carry a camera with you so that if you do find something inspiring you can take a few snapshots of it and store it on your computer for reference. Other sources of inspirations include movies and television but also books and good literature. Many game development studios also usually employ concept artists who create illustrations and sketches of the games environments that are passed on to the production artists for inspiration.

How do you work and in what order?

As mentioned earlier, the first step is to find as many good references as possible of what I'm about to create. If I'm working with a 3D modeler I will usually wait for him to finish up the 3D model. I then unwrap it and start the texturing process with the help of my reference material. As I am working on the textures I do my best to get feedback from the art director to make sure I'm on the right track. Once the texture is completed I prepare for importing everything into the game engine. I will then import the assets and, depending on the complexity of the rendering system, create the necessary shaders from my textures. Some tweaking may be necessary here and there once the object is in the game, so I often go back and forth for a little while to make those changes. Once I am satisfied I move on to the next object or area.

Why do you make certain decisions? What is your philosophy and what do you do to try to achieve it?

When doing game art, one very important thing is to not regard every asset one works on as a single asset but as part of a larger whole. After all, every object is placed in the world alongside other objects and the whole of that is what makes the environment what it is. For instance you can make the most awesome, most detailed looking truck but if it doesn't fit in well with the rest of the environment it is basically useless. This takes a lot of practice and learning-by-doing. At the end of the day though, the payoff is huge! You know you have done well if you look at your environment, nothing looks out of place and everything "works" and looks like it really belongs to the same environment.

What do you do to maintain the consistency between different elements and the rest of the environment? How do you preserve the unity of the world?

One good way of doing this (in regards to textures and color balance) is to try and unwrap your object so all the textures fit on a single map. That way it is very easy to evenly balance out colors and details on all the pieces. Sometimes you cannot do this due to the way the model is built, engine-, memory-, or streaming restrictions. What I do in this case is create a big blank canvas that I paste all my textures into so that I can see them side-by-side and tweak them with the "bigger picture" in mind. One can do similar things for the actual 3D environments. Start placing objects in your scene, light it, then go in-game to see if everything works together. Be critical with yourself and your work!

How do you develop a certain style or art/game direction for your aspects in the world?

As an example: if one has worked on realistic textures for a long time and is suddenly asked to make cartoon art on the next project it will require a totally different approach. Here is why.

When I create realistic textures I subconsciously work by a set of “rules” that I have in my mind but that are not written down. For example, when I’m working on a brick texture there are a lot of moments where I’m thinking “OK, now do this”, or “don’t do that”, with the idea in mind that the end result is supposed to look realistic. Every brush stroke I make either adds to the realism or takes away from it.

When I need to change to a new style I have to throw all these rules out the window and come up with new ones that nail down this “new style” I’m working in. The best way to nail it down is to make “sample textures”. A set of materials that I know for sure are going to appear in the game. I talk to my art director and show him my tests and get feedback on them. Then I go back and do more tests with different materials until I have clearly defined the new style.

What’s the most commonly made mistake?

A common mistake is that many people add too much grime and dirt to their textures and really overdo it instead of keeping the dirt relatively subtle and sparse and only placing it in spots where it would make sense. I used to do the same thing. I felt like every texture I was making had to have lots of dirt in it just so it doesn’t look boring. Then I realized that a texture is way more interesting if you treat details the way they are meant to be treated...as details. Think about it like salt and pepper in your soup. If you don’t have enough it’s bad, if you add too much it’s bad, too. So you want to find the golden middle.

Another mistake is the use of bad source photos. Too often have I seen textures made up of low-res and pixelated photos with little to no touch-up work done to them. If you use a photo source (on that note: there’s nothing wrong with it, everybody does it!) make sure that it’s a good photo source. I do prefer a nice handpainted texture over a bad photo texture any day though.

What should people always keep in mind?

When making textures one really has to stay objective and self-critical about their work. If one is not sure about a change they’ve made to one of the textures and find themselves hitting “undo” and “redo” over and over again, chances are one probably doesn’t like it and is trying to convince themselves that they do. When that happens it’s usually the best idea to go with the original. Or in case of serious doubt get feedback from somebody else.

Pay close attention to subtleties in the world and keep them in mind when doing artwork. Get feedback from other people. Even encourage people to be harsh with your work when they critique it. Because, at the end of the day we learn more from our mistakes than from what we do right.

MATTHEW FLORIANZ - SOUND

Matthew Florianz, born and raised in the Netherlands, has been working in the games industry since 2001. As a graphic designer, his first assignment was that of level designer on several shooters. During these projects he also doubled as an audio producer and currently works as the full-time audio director for the MMORPG *The Chronicles of Spellborn*.

His interest in sound comes from his background in ambient music (www.matthewflorianz.com) which he has been producing since 1995. For *The Chronicles of Spellborn*, Matthew has been a liaison to music-composer Jesper Kyd and was responsible for all in-game sounds and background ambiances.

How do you begin?

Before running off and tackling problems head on, it's better to start working on something slowly. If you can work on more than one "project" it's good to start thinking about something that you're not immediately working on; let the ideas and background information stew for a while in your mind.

This gives you the time and room to come up with something original.

Where do you find your inspiration?

Inspiration is an odd thing to describe, let alone find. When creating sounds, ambiance, or music the choices made are related to what I liked when I was much younger and far more impressionable. Seeing a film with great atmospheric sound (*Alien*) or visiting a spooky castle in a theme park are impressions which have left their traces.

One example that always comes to mind is the single player campaign from *Unreal*. Early on in the game the player enters a mine and cavern system and quite some time is spent underground. When the player finally emerges from the underground, the sun is just setting and warm ambient music plays back above ground. The tunnels, on the other hand, were full of deep drones and basses. I am not even sure if it all really was that long, and if the underground was as dark as described, but it felt like it, and in no small part thanks to a minimal but very efficient sound design.

How do you work and in what order?

I start outside the editor, and even outside the world that's inside it. Instead, I want to try and think about what's going on in very broad terms. Instead of thinking about trees and plants, I think about the forest, the colors, the temperature, and mood. These all go hand in hand suggesting what sounds to use, and in what rhythm to use them.

A streaming ambiance (much like music) results from this. It captures the mood, the essence of an environment in sound that is not related to the details of an environment, but its general mood.

When the ambiance is set, then locational sound (effects) are placed. Specific objects such as the leaves on a tree or water running in a stream are given their own sound. If done right, the effects and the ambiance will melt together and create a rich and full soundscape.

Why do you make certain decisions? What is your philosophy and what do you do to try to achieve it?

In movies there is a saying that music should be felt, not heard. It expresses a work ethic in which the composer does not try to show off their extraordinary writing skills; instead they apply them to the canvas, so to speak. Good film music balances between being very “visible” where it needs to be, and guiding emotions in a more subtle way where it can. It’s a dying art form, unfortunately, and the same goes for sound design. In my opinion sound should not over-act, it should never make you aware that you are listening, because the moment it does communicate directly with your senses, you run the risk of breaking the spell that a movie, or good game casts. It’s often called immersion and when a sound suddenly flies around your head, all it really does is take your eyes from the screen and make you aware of having a head. A bit over exaggerated but a good dogma to follow nonetheless, which doesn’t mean it cannot be broken.

I tend to dislike graphic-intense games where, for instance, blooming or bump-maps have been used too obviously. I prefer games that downplay these technical achievements, treat them not as something special, but as tools to help suspend a player’s disbelief. No audiovisual element should stand out so strongly that it can take the player out of the game experience. The best sound in my opinion is the sound one notices when it’s gone but doesn’t when it’s there.

Still, I am very aware that players may eventually start to pay specific attention to the atmospheric sound and as much as I have to make sure that nothing stands out, on the other hand, when picked out it also has to offer something special. One way to achieve this is by adding abstract sound stories. The movie *Seven* is great material for study in this respect, there’s always something going on outside the windows of apartments, someone having an argument, noises of fights or other unpleasantness. The movie also has a fantastic explanation of the music and sound design, and I think it would explain a great deal more about sound than I could.

What do you do to maintain the consistency between all the different elements and the rest of the environment? How do you preserve the unity of the world?

By listening and listening and then listening once again, there’s really no other way to do it. It helps to listen in an unfamiliar setting. Use headphones when you always work on speakers, or the other way round. It helps you focus not on the work you did, but the way it ended up in the environment. It’s very hard to look or listen to something you made and be a listener instead of a creator. But being able to listen to your own work is hugely important and not as easy as it might seem.

How do you develop a certain style or art/game direction for your element in the world?

Probably by just working and listening critically for what works and what doesn’t. Making games is a team effort and when something I have created doesn’t work, I’ll hear about it quickly. Of course, my background in ambient music helps and there is cross-pollination from doing sound for games and sound for albums. Often the game’s style is already there in the visuals; the process of creating atmospheric sound is as simple as connecting the dots in such a case.

Which aspects are most important to you? What part of the process or element do you devote the most time to?

Most of my time is spent tweaking volumes and making sure that nothing draws attention to itself more than it should. Of course, effects such as weapon effects or interaction sounds are different from the background ambience, but taking this into consideration, sound can quickly become too much or annoying if it doesn't work well with the visuals. If the sound is bad, it is most often the tone and volume that make it so.

What's the most commonly made mistake?

A mistake often made is sound for the purpose of having sound. The same goes for music and voices. What we did for a certain project is to hire stage performers instead of voice actors. Although their voices don't have the strength and clarity of a voice actor, they do bring something into their performance which I find far more interesting than having the perfect sounding voice. "What can the sound add?" should always be a sound designer's principle question, not; where one can add more sound.

What should people always keep in mind?

Wherever you can, integrate the sound with the environment and don't try to take the environment too literally. Sometimes completely abstract sounds or noises can create an interesting counterbalance to the images. Compose the sound in your worlds as if it is a symphony, where the players movement speed is akin the tempo in music.

Strengthen contrasts in areas such as cold or warm, harsh or gentle, soft or loud. Games are not very good in working with all your senses but you can definitely try and bring them out more through sound, and of course visuals.

Blue feels cold, so don't use a droning bass sound in such an environment, use higher pitched noises. Then again, for every such rule or observation, there is a good occasion to purposely break away from it.

LIST OF SCREENSHOTS

• Cover

Screenshots of Krodan for UT3, The Spire from Gears of War, both custom levels, and a random corridor. Created by Sjoerd De Jong using Epic Games Inc. Unreal ® editing software. Gears of War is ©2006 Epic Games Inc. Used by Permission.

• Chapter Pages

All chapter pages use screenshots from Horean, a custom Unreal Tournament DOM level. Created by Sjoerd De Jong using Epic Games Inc. Unreal ® editing software.

• Gameplay (GP)

1-8 and 10-14 Created by the author using Epic Games Inc. Unreal ® editing software.
9 CTF-Coret - Unreal Tournament - ©[1999] Epic Games, Inc.
15 City 17 - Half Life 2 - ©[2004] Valve Software.

• Audiovisuals (AV)

1-4, 17-18, 53-54, 69-70, 74 Created by the author. ©[2008] Sjoerd De Jong.
5-9, 12-16, 20-29, 33, 35 (L), 42-43, 48-49, 52, 55 (R), 56, 58-65, 67, 71, 73, 75-77 Created by Sjoerd De Jong using Epic Games Inc. Unreal ® editing software.
30-31, 34, 41, 44, 45-47, 50, 65, 72 Created by Sjoerd De Jong using Epic Games Inc. Unreal ® editing software and art assets.
32, 36-38, 51, 57, 78-79 - ©[2008] Spellborn Works NV.
35 (R) ONS-Adara - Unreal Tournament 2004 - ©[2004] Epic Games, Inc.
55 (L) DM-Idoma - Unreal Tournament 2004 - ©[2004] Epic Games, Inc.
66 Operation Na Pali - Unreal Tournament - Created using Epic Games Inc. Unreal ® editing software.
10-11 The Elder Scrolls IV - Oblivion - ©[2006] Bethesda Softworks/Zenimax.
19 DM-Rankin - Unreal Tournament 2004 - ©[2004] Epic Games, Inc.

• Examples (EX)

1-19 Created by Sjoerd De Jong using Epic Games Inc. Unreal ® editing software.

CREDITS AND THANKS

- Matt “ArcadiaVincennes” Bromley – Editor
- All of my interviewees – for helping me out!
- Frank Bakker – Website code
- The Unreal Community – for being cool.
- The authors of the levels I used as examples.
- Epic Games – For allowing me to use the Unreal Engine and games for all the example content.
- Spellborn Works and Khaeon Games – For the TCOS content.

Additional Help

- Mario “nELsOn” Marquardt
- Anton Botvalde
- Maxx “Errol” Wyndham
- David “DavidM” Munich
- Rogelio “Desperado2” Olguin
- Peter “Dynamit” Boström
- Tom “Gmotha” Goetschalckx
- Katrien Anthonissen

And any one else who helped me accomplish this book.

COPYRIGHTS AND PERMISSIONS

This book “The Hows and Whys of Level Design – Second Edition” is copyright 2008, by Sjoerd “Hourences” De Jong

By possessing and/or reading this book you agree to the following usage terms.

- You may not reprint, republish, alter, translate or reproduce this book or any part of it in any way and in any language without my explicit written permission.
- You may not charge any money for this book or any of its embedded content. Only I am allowed to do so.
- You may not distribute this book or any of its embedded content for free: neither offline nor online.
- You must seek my written authorization if you would like to use parts of this book's content for publication on a website, magazine, school work, or other such means of distribution.

I can be contacted by email at book@Hourences.com

I am an independent individual who is in no way supported by any company, group, or organization. I put a lot of work and personal knowledge into this book. By stealing content or illegally reproducing this book you aren't hurting a multi-million dollar company. You hurt me as the author. Please respect my work. If you obtained this book through illegal means, please buy it if it proves helpful to you. Knowledge is precious and if it can get you a (better) job then those few dollars are well worth the money.

All images are the property of and copyrighted by their respective owners.

Epic Games, the Epic Games logo, Unreal, Unreal Engine, Gears of War, the Crimson Omen logo, Unreal Tournament and the Unreal tournament logo are either registered trademarks or trademarks of Epic Games, Inc. in the United States and/or elsewhere.

The Chronicles of Spellborn (c)2008 Spellborn Works, NV. All Rights Reserved. The Chronicles of Spellborn logo and name are registered trademarks of Spellborn Works, NV.

The Elder Scrolls: Oblivion (c)2006 Bethesda Softworks, LLC, a ZeniMax Media company. All related logos are registered trademarks or trademarks of ZeniMax Media Inc. in the U.S. and/or other countries. All Rights Reserved.

Half-Life (c)1998 and Half-Life 2 (c)2004 Valve Corporation. All rights reserved.

All other trademarks and trade names are properties of their respective owners.

Any level displayed in the images is the property of its respective owner.

Some level images may display an altered or incomplete version of the level in question. Any alterations were made for the sole point of creating examples.

